Paraphrase Identification Using Knowledge-Lean Techniques

Asli Eyecioglu Ozmutlu

Submitted in partial fulfilment of

the requirements for the degree of

Doctor of Philosophy

In the Department of Informatics and Engineering,

University of Sussex

November 2016

Abstract

This research addresses the problem of identification of sentential paraphrases; that is, the ability of an estimator to predict well whether two sentential text fragments are paraphrases. The paraphrase identification task has practical importance in the Natural Language Processing (NLP) community because of the need to deal with the pervasive problem of linguistic variation.

Accurate methods for identifying paraphrases should help to improve the performance of NLP systems that require language understanding. This includes key applications such as machine translation, information retrieval and question answering amongst others. Over the course of the last decade, a growing body of research has been conducted on paraphrase identification and it has become an individual working area of NLP.

Our objective is to investigate whether techniques concentrating on automated understanding of text requiring less resource may achieve results comparable to methods employing more sophisticated NLP processing tools and other resources. These techniques, which we call *"knowledge-lean"*, range from simple, shallow overlap methods based on lexical items or n-grams through to more sophisticated methods that employ automatically generated distributional thesauri.

The work begins by focusing on techniques that exploit lexical overlap and text-based statistical techniques that are much less in need of NLP tools. We investigate the question "To what extent can these methods be used for the purpose of a paraphrase identification task?" For the two gold standard data, we obtained competitive results on the Microsoft Research Paraphrase Corpus (MSRPC) and reached the state-of-the-art results on the Twitter Paraphrase Corpus, using only n-gram overlap features in conjunction with support vector machines (SVMs).

These techniques do not require any language specific tools or external resources and appear to perform well without the need to normalise colloquial language such as that found on Twitter. It was natural to extend the scope of the research and to consider experimenting on another language, which is poor in resources. The scarcity of available paraphrase data led us to construct our own corpus; we have constructed a paraphrase

corpus in Turkish. This corpus is relatively small but provides a representative collection, including a variety of texts. While there is still debate as to whether a binary or fine-grained judgement satisfies a paraphrase corpus, we chose to provide data for a sentential textual similarity task by agreeing on fine-grained scoring, knowing that this could be converted to binary scoring, but not the other way around. The correlation between the results from different corpora is promising. Therefore, it can be surmised that languages poor in resources can benefit from knowledge-lean techniques.

Discovering the strengths of knowledge-lean techniques extended with a new perspective to techniques that use distributional statistical features of text by representing each word as a vector (word2vec). While recent research focuses on larger fragments of text with word2vec, such as phrases, sentences and even paragraphs, a new approach is presented by introducing vectors of character n-grams that carry the same attributes as word vectors. The proposed method has the ability to capture syntactic relations as well as semantic relations without semantic knowledge. This is proven to be competitive on Twitter compared to more sophisticated methods.

Keywords: *Paraphrasing, Knowledge-Lean, Twitter, Turkish, MSRPC, SVMs, N-grams, Overlap methods. Word2Vec*

Thesis supervisor: Dr. Bill Keller

Declaration

I declare that this thesis was composed by myself, that the work contained herein is my own except where explicitly stated otherwise in the text, and that this work has not been submitted for any other degree or professional qualification except as specified.

Asli Eyecioglu Ozmutlu

Contents

List of Tables

List of Figures

Abbreviations

A-O

DSMs: Distributional Similarity Models
FAQs: Frequently Asked Questions
IE: Information Extraction
ML: Machine Learning
MSRPC: Microsoft Research Paraphrase Corpus
NLP: Natural Language Processing
NNM: Neural Network Models
OOV: Out-of-vocabulary

P-R

PAN: Plagiarism Detection Corpus
PE: Paraphrase Extraction
PG: Paraphrase Generation
PI: Paraphrase Identification
PoS: Part-of-Speech
QA: Question Answering
RBF: Radial Basis Function
RNNM: Recurrent Neural Network Models

S_Z

STS: Semantic Textual Similarity
SVC: Support Vector Classifier
SVM: Support Vector Machine
TE: Textual Entailment
TPC: Twitter Paraphrase Corpus
TuPC: Turkish Paraphrase Corpus
VSM: Vector Space Model

Acknowledgments

I acknowledge the contributions of the following beloved people.

"I would like to express my sincere and deep gratitude to my supervisor, Dr. Bill Keller." And I would like to paraphrase the previous sentence in so many ways, to show how grateful I am to have had such a pillar of strength during my project. Your knowledge and guidance form an endless corpus of wisdom that I will keep for my whole life!

I also sincerely thank my MSc supervisor, Dr. Johann Briffa, whose guidance helped me to start my PhD, and my thesis committee members Prof. John Carroll and Dr. Paul Newbury for their guidance and enlightenment each year at my annual review.

I am fortunate to have the following people as my friends and to work with them in the same lab: Grecia, Ronald, Eltayyib and Rosanno. The advice and help they gave whenever I needed it gave me strength through the more difficult times.

I am thankful to my other half, Bekir, who has filled my life with love and support. Your patience and support towards me is incredible.

My beloved parents, Celal and Leyla, have enhanced my world with their endless love. To my sisters, Dilek and Hilal, my cousin Bengisu, and my niece Asya: your presence in my life has been irreplaceable; life could never be joyful without you.

I am fortunate that my world is surrounded with people who helped me stay sane through difficult times: Arzu, Fahad, Shuna, Mevra, Rabia, Leyla, Tugba, Ozden, Yasemin.

I greatly value the existence in my life of so many more people, too numerous to be able to name here.

I deeply appreciate the above people, and their belief in me.

Chapter 1

1 Introduction

1.1 Motivation

This thesis is concerned with natural language processing (NLP) techniques for paraphrase identification. According to Lintean and Rus (2011), paraphrase identification may be defined as "the task of deciding whether two given text fragments have the same meaning". It is also known as paraphrase detection (Socher, Huang, Pennington, Ng, & Manning, 2011; Zhang & Patrick, 2005) or paraphrase recognition (Androutsopoulos & Malakasiotis, 2010). Paraphrase identification methods simply take a pair of language expressions and make a judgement as to whether they are paraphrases.

Although the Paraphrase Identification (PI) task aims to identify sentences that are semantically equivalent, a number of researchers have shown that classifiers trained on lexical overlap features may achieve relatively high accuracy. Good performance is achieved without the use of knowledge-based semantic features or other external knowledge sources such as parallel corpora (Blacoe & Lapata, 2012; Lintean & Rus, 2011). This thesis is concerned with methods for paraphrase identification that can be considered *knowledge-poor aka knowledge-lean*. The term "*knowledge-poor"* is first introduced (as far as we are aware) by Hearst and Grefenstette (1992). Their motivation is the usage of knowledge-poor corpus-based approaches for the automatic discovery of lexical relations. They believe that knowledge-poor corpus-based approaches result in stronger results without the need of complex knowledge-based approaches. The term "knowledge-lean" is used in the same way by Pedersen and Bruce (1998) in order to draw attention to the significance of corpus-based measures compared to knowledge-based measures for word sense disambiguation task.

A knowledge-lean approach is a strategy that requires less alteration of experimental data, less usage of time-consuming resources, and less complex methods: briefly, it aims to reduce transformation of data considerably. This prevents losing useful information, which is usually thrown out or lost during transformation process. Despite the fact that these methods are considered shallow, their performance can surpass that of more elaborate approaches.

Knowledge-lean approaches have better potential applicability to less resourced languages, reduction in manual annotation effort, and potential to learn from current samples of the actual data to be processed rather that a possibly unrepresentative approximation. Methods are considered knowledge-lean if they make use only of the text at hand, and avoid the use of external processing tools and other resources. This approach has led the NLP community to the usage of techniques that require less knowledge, in order to avoid the costly and time-consuming construction of knowledge-rich resources. Knowledge-lean PI methods may thus employ shallow overlap measures based on lexical items or n-grams, but they might also make use of distributional techniques based on simple text statistics.

During the last two decades, paraphrase identification task has gained significant importance in applied Natural Language Processing and Computational Linguistics. The paraphrase identification task has practical importance in the NLP community because of the need to deal with the pervasive problem of linguistic variation. In particular, two NLP tasks, Textual Entailment (TE) and Semantic Textual Similarity (STS), are associated with the Paraphrase identification task. Madnani and Dorr (2010) show the difference between PI and TE with the sample sentences S1, S2 and S3 (Table 1-1). S1 entails S2 and S3, but these two entailment sentences S2 and S3 are not paraphrases of the text S1. We also added another sentence, S4, in addition to this example. The sentences S4 and S2 have a bidirectional relation so that they are paraphrases. Among these four sentences, they certainly have similarity that can be graded for the STS task, whereas PI and TE use the binary criteria (1 if there is a relation between the two text fragment; 0 otherwise).

Task	Relation	Decision	Sample Sentences
TE	Unidirectional	Yes/No	S1: Yahoo's buyout of Overture was finalized S3: Overture is now owned by Yahoo
PI	Bidirectional	Yes/No	S2: Yahoo bought Overture S4: Overture was bought by Yahoo
STS	Bidirectional	Degree (0-5)	S2: Yahoo bought Overture S1: Yahoo's buyout of Overture was finalized.

Table 1-1: The differences among textual entailment, paraphrase identification and semantic textual similarity tasks

Sentence pairs are considered to have a bidirectional entailment relation, which increases the coverage of Textual Entailment systems. Dagan (2008) proposes a concrete semantic engine for textual entailment tasks; proposing that this semantic engine also searches for possible paraphrase relations, if the two entailed sentences have semantic equivalence. Lately, Semantic Textual Similarity tasks have been integrated with paraphrase identification. Sentence pairs are assigned a degree of semantic equivalence instead of having a binary score. Recent STS shared tasks on sentence similarity (Agirre, Cer, Diab, & Gonzalez-Agirre, 2012; Agirre, Cer, Diab, Gonzalez-Agirre, & Guo, 2013) and also on multilingual sentence similarity (Agirre et al., 2014, 2015) present a great quantity of research.

Development of various other NLP applications might benefit from paraphrase identification methods. Question Answering (QA) systems deal with cases where the answer to a question might not always be in the same form as the question. The alternative answer can be retrieved from a text by rephrasing the sentence. Producing several variants of a question also increases the possibility of finding the right answer to the question. Mckeown (1983) addresses solutions to the problems in natural language systems by using a paraphraser mechanism. Ravichandran and Hovy (2002) explore paraphrased patterns in order to improve QA systems. Barzilay, Mckeown, and Elhadad (1999) show that paraphrases increase the chance of finding the right brief statement of a text for a summarization task of multiple sources such as news articles. Different phrases explaining the same event create alternatives for generating a summarized sentence from a text.

Moreover, Statistical Machine Translation (SMT) is one of the areas that adjusting few rules to a translation algorithm can increase its performance (Madnani, Ayan, Resnik, Dorr, & Park, 2007; Owczarzak, Gorves, Genabith, & Way, 2006). Finding paraphrase pairs supplies more words, phrases and sentences to be translated. Therefore, the redundant data from the translation process can be utilized using paraphrasing tasks (See Section 3). SMT systems are trained on 'bitexts', bilingual parallel corpora (Bannard & Callison-Burch, 2005); augmenting such data with paraphrases can significantly improve translation quality and coverage (Callison-Burch, Koehn, & Osborne, 2006; He, Zhao, Wang, & Liu, 2011).

Several recent attempts to the use of paraphrases to augment the coverage of SMT include filtering out the out-of-vocabulary (OOV) words (Marton, Callison-Burch, & Resnik, 2009) and identifying particular words such as negators and antonyms (Marton, Kholy, & Habash, 2011) when evaluating SMT applications. On the other hand, SMT metrics, alone and in combination, can be used to identify sentence-level paraphrases (Finch, Hwang, & Sumita, 2005; Madnani, Tetreault, & Chodorow, 2012; Owczarzak et al., 2006).

Information Retrieval (IR) and Information Extraction (IE) methods are mostly used in combination with to other NLP methods and may benefit from paraphrase patterns. An early paper on IR argued for the utility of paraphrase (Culicover, 1968) and paraphrases of words in IE patterns can be identified so as to extract the required information from stored text (Shinyama & Sekine, 2003).

Barron-Cedeno et al. (2013) address the difficulty of detecting paraphrases for an automatic plagiarism detection system. They analyse the relationship with paraphrases and plagiarised texts and suggest that identifying paraphrases improves the performance of plagiarism detection systems. Ganitkevitch et al. (2011) present state-of-the-art results for compression systems, and benefit from the extraction of sentential paraphrases. Also, Natural Language Generation (NLG) Systems may benefit from paraphrasing for sentence re-writing task (Power & Scott, 2005).

Paraphrase Identification task has attracted interest in the NLP community and recently it has become an individual working area of NLP. A growing body of research has been conducted and various semantic and syntactic tools and resources have been proposed for the problem of identifying paraphrases (Chapter 2). However, the knowledge-rich approaches may be inadequate for capturing language variation, because they are limited by manually constructed resources. A knowledge-lean approach, on the other hand, has no limits or binds to tools or time-costly resources, which makes it more suitable for a wide range of applications. There is still a debate as to whether the machine can learn without deeper processing and semantics, and moreover, whether similar results can be obtained without using knowledge-rich techniques and resources.

This thesis investigates whether techniques concentrating on automated understanding of text, requiring less resource may achieve results comparable to methods employing more sophisticated NLP processing tools and other resources. Discovering the strengths of knowledge-lean techniques by comparing the features of identified paraphrase pairs is the main objective. Therefore, knowledge-lean methods will be examined to see to what extent these techniques can be used for the purpose of paraphrase identification tasks.

The research reported in this thesis aims to examine knowledge-lean approaches due to their applicability to various types of texts. Even widely used, knowledge-rich, sources, such as WordNet (Fellbaum, 1998), are not complete. In WordNet, each lexical item is grouped into major category tags (Verb-Noun-Adjective-Adverb) and there is a conceptual similarity defined based on synonymous relations such as hyperonymy, hyponymy and etc. The relation between each two items defined with a similarity score. These similarities (Pedersen, Patwardhan, & Michelizzi, 2004) are computed by measuring the distance between two words. However, it is difficult to define word similarities in terms of WordNet distances.

In essence, knowledge representation is a strategic problem. Therefore, Halevy, Norvig, and Pereira (2009) claim that the data no longer require processing primarily. They propose that a large amount of data can be gathered from the web using machine learning techniques. This obtained data might overcome more limitations where there is not enough annotated data.

Rus, Banjade and Lintean (2014) argue that the best methods for the paraphrase identification task can be obtained from lexical overlap measures, optimized with the combinations of several pre-processing steps. We start by exploring the question *"How effective is it to use pre-processing with knowledge-lean techniques that are based on simple overlap measures, in order to identify paraphrase pairs?"* We show that by investigating the overlapping features with the various representations of experimental data in Chapter 3, then, we query the usage of pre-processing techniques and explore the combinations of overlap features on raw data in Chapter 4.

Since knowledge-lean techniques do not require any semantic or syntactic tools, the question becomes *"Can knowledge-lean techniques be adapted to another language for paraphrase identification?"* The limitation of paraphrase corpora becomes another issue in answering this question, which leads us to construct a paraphrase corpus in another language. Chapter 5 reports the detail of a newly constructed Turkish Paraphrase Corpus, and we examine previously used techniques for identifying Turkish paraphrase pairs.

We expect to find the extent of the usage of knowledge-lean techniques for paraphrase identification beyond relying on overlapping features. We ask: *Can continuous word representations help to identify semantic relations between a pair of sentences without usage of semantic tools?* In addition: *Will this help us to identify paraphrase pairs?* The first experiment to explore this uses distributional similarities of words using simple overlap features, in Chapter 3. In Chapter 6, one recent Neural Network approach – word2vec– (Mikolov, Corrado, Chen, & Dean, 2013) that underlies the distributional hypothesis is used for sentence representation, by capturing the semantic relations between word pairs from a very large unlabelled data. Next, we extend the word2vec approach by capturing the relations of character n-grams.

1.2 Outline of the Thesis

In the next chapter, we provide a rich literature review of paraphrase identification methods and focus on the research conducted on paraphrase identification in relation to the applied knowledge-lean techniques. A substantial background of paraphrasing applications will be provided, taking into account the considerable research related to this area.

Chapter 3 sets up the basis for the exploration of knowledge-lean approaches by exploiting simple overlap measures with reference to data representation techniques, which are then computed for two different paraphrase corpora. These shallow overlap methods exploit character and word n-gram features, which then computed with a few widely accepted similarity measures. These features will be elaborated also in Chapter 4. Later in Chapter 3, distributional lexical similarities from a small distributional thesaurus are used for sentence representation of paraphrase pairs. The findings from this experiment provide an insight into the usage of distributed sentence representation with vectors, given in Chapter 6.

In Chapter 4, a colloquial paraphrase corpus, Twitter Paraphrase Corpus (TPC), is used to experiment with various shallow methods for identifying paraphrases that are based on lexical and character n-grams. A SVM classifier using linear and radial basis function kernels is used for classification. Finally, we discover a set of stable features used in our system, the results of which out-performed many sophisticated methods without use of any external tools or resources. These features are extracted for the previous paraphrase dataset, Microsoft Research Paraphrase Corpus (MSRPC), and show competitive results. We conclude with a comparison of the overall results.

Chapter 5 is motivated by the competitive results obtained in Chapter 4. The MSRPC is a collection of news articles, whereas TPC is constructed from a collection of non-literarily written text. This raises the query of whether knowledge-lean techniques are applicable to other languages. We take this research one step further by exploring the applicability of knowledge-lean methods in another language. While paraphrase identification task is newly gaining importance in NLP, there are a few paraphrase datasets in other languages such as Dutch, French etc., but they were not specifically built for paraphrase identification purpose. In consideration of the lack of paraphrase corpora, we contribute by building a paraphrase corpus in Turkish. The resulting Turkish Paraphrase Corpus (TuPC) is used to experiment with our best performing methods from Chapter 4. The results are evaluated in comparison to the TPC and MSRPC results.

In Chapter 6, a recent popular approach of Neural Network Models that are built based on continuous representation of words as vectors –w2v– is used to experiment with

the two paraphrase corpora: TPC and MSRPC. Capturing semantic regularities of text from very large unstructured and unlabelled data enriches the knowledge-lean methods that are previously used to experiment with overlap measures. Moreover, a new approach, vector of n-grams –ng2v– motivated by the w2v models is presented. Additionally, we explore whether w2v and ng2v statistical models trained on a small dataset perform well in cases where data sparseness is the problem.

Chapter 7 includes a summary of our findings throughout this thesis. It reviews the objectives that motivate the research questions, specifying the contribution of this thesis to the area of paraphrase identification. We will draw attention to future directions in considering effective usage of knowledge-lean methods for the paraphrase identification task.

A table of system names used throughout the thesis is also provided in Appendix A (Table 0-1). The table shows data requirements, features and measures used in each individual chapter.

Chapter 2

2 Paraphrase Identification and Related Research

2.1 Introduction

This chapter will present an overview of work on paraphrase identification and related tasks.

We begin first by considering the definition of the notion of a ‘paraphrase’ and then explain the previously used corpora in paraphrasing applications.

The following part of the chapter sets a line of demarcation among paraphrase tasks in terms of the way paraphrasing methods are performed; *paraphrase identification, paraphrase generation and paraphrase extraction*. The survey of Androutsopoulos and Malakasiotis (2010) proposes a clear distinction between each paraphrasing tasks without ignoring their intersecting relationships. Indeed, the boundaries between paraphrasing tasks are not strictly indicated by the most published research. One reason is that different paraphrasing tasks may be combined in ways that make it hard to distinguish between them. For instance, a system can extract new paraphrases from those generated, exploiting both paraphrase extraction and generation techniques (Bannard & Callison-Burch, 2005; Barzilay & McKeown, 2001; Callison-Burch, Cohn, & Lapata, 2008; Madnani et al., 2007). Another reason is that the problems are often illustrated with NLP tasks that induce paraphrases eventually, so that specifying the paraphrasing tasks becomes secondary issue. It is hard to draw a line while paraphrasing tasks has been explored as part of the other NLP applications. In particular, the paraphrase identification (PI) task has become central to many NLP problems, such as sentential semantic similarity, textual entailment, summarization, sentence compression etc. We then examine the research conducted on

paraphrase identification methods. Later, a brief explanation of the various methods of paraphrase generation and paraphrase extraction is provided.

2.2 Definition of paraphrasing

Table 2-1 shows a variety of paraphrasing definitions suggested by different authors. This table gives the best understanding by showing that paraphrasing is more complex than simply replacing words with their synonyms. The nature of a paraphrase is also clarified in Hirst (2003) by indicating that synonymous relations are not enough to describe paraphrasing.

	Definition	Author
Paraphrasing	is generally considered to be a meaning-preserving relation	(Culicover, 1968)
	alternative way to convey the same information	(Barzilay & McKeown, 2001)
	represents (possibly partial) equivalencies between different expressions that correspond to the same meaning	(Glickman & Dagan, 2003)
	is talking about same situation in different words and different syntax	(Hirst, 2003)
	is the restatement (or reuse) of text giving the meaning in another form	(Fernando & Stevenson, 2008)
	is a text-to-text relation between two non-identical text fragments that express the same idea in different ways	(Lintean & Rus, 2010)
	is semantic equivalence	(Madnani & Dorr, 2010)
	is the act of replacing linguistic utterances (typically text) with other linguistic utterances, bearing similar meaning but different form.	(Marton, 2010)

Table 2-1: Paraphrases of Definition of Paraphrasing

Culicover (1968) appears to have been the first to provide a definition of, and computational approach to, paraphrasing problems. Many of the types of paraphrases stem from his proposals. He states that the definition of paraphrasing, ‘same in meaning’, is not illuminating in the absence of human intuition. Hence, he proposes several paraphrase types, considering their grammatical, transformational and lexical relations.

The process of paraphrasing could be understood as first taking words individually, and then using phrases to combine them as sentential paraphrases in terms of the applied methods. This process is likely inductive. The inductive process has mainly three aspects for the purpose of analysing the paraphrase approaches. The first two are the syntactic and lexical variations of the language. The third is just a synthesis of the first two approaches.

In previous research, syntactic and lexical relations were examined separately depending on application type: whether it requires lexical or syntactic benefits for a specific application. To increase the efficiency of a system, recent research has been led to take advantage of both the syntactic and the lexical variability of language. Moreover, paraphrasing can also involve real-world knowledge, e.g. President of France ↔ Charles de Gaulle (Culicover, 1968).

Paraphrase pairs are often categorized as lexical, phrasal or sentential paraphrases (Madnani & Dorr, 2010). Linguistically, lexical paraphrases are considered according to their relationships such as synonymy, hyponymy, and hyperonymy. However, Hirst (2003) claims that paraphrases mostly correspond to the pragmatic differences in meaning, such as near-synonyms, connotations, and implications rather than absolute synonymous relations. Moreover, he characterises paraphrases as large-scale paraphrasing or small-scale paraphrasing (Figure 2-1). Large-scale paraphrasing addresses the different emphasis in sentences by using syntactic variations. This type of paraphrase is evaluated according to the stance, attitude, and opinion of a speaker. Small-scale paraphrasing is defined as lexical nuances.

Large Scale Paraphrasing	***At least 13 people were killed*** by a suicide bomber on a bus in downtown Jerusalem this morning.	***Primary emphasis in main subject and verb***
	A suicide bomber blew himself up on a bus in downtown Jerusalem this morning, ***killing at least 13 people.***	***Secondary emphasis in appositive clause***
Small Scale Paraphrasing	The President ***addressed*** the nation.	***More formal***
	The President ***spoke to*** the nation.	***Less formal***

Figure 2-1: Examples of large and small scale paraphrasing (Hirst, 2003)

Phrasal paraphrases concern paraphrase relations between sub-sentential constituents, that is constituent parts of a sentence such as noun-phrases, verb-phrases and clauses. Pattern variants are also accepted as phrasal paraphrases. In the example below, S^1, S^2 and S^3 are paraphrases of one another. X and Y slots can be replaced with different words (Androutsopoulos & Malakasiotis, 2010):

S^1: X wrote Y.

S^2: Y was written by X.

S^3: X is the writer of Y.

Sentential paraphrases require a whole sentence to be replaced with another one with a different form, both sentences preserving the same meaning. Most of the recent paraphrase identification research considers sentential paraphrases. Sentential paraphrase identification is regarded as a challenging task because of the semantic equivalency requirement, which is conditioned by the context between the two sentences.

2.3 Corpus Data for Paraphrasing Tasks

To date, several types of corpora have been used for different paraphrasing problems: Single Monolingual Corpora, Monolingual Comparable Corpora, Monolingual Parallel Corpora, and Bilingual Comparable Corpora (Madnani & Dorr, 2010). Corpus types play a significant role in the development of paraphrasing methods, which will be explained respectively according to their improvement.

2.3.1 Single Monolingual Corpora

Early paraphrasing methods generally utilised single monolingual annotated corpus by employing distributional similarity methods in order to extract paraphrases. Although this type of corpus is helpful on finding paraphrasing patterns on lexical and phrasal level, the limited quantity of data can cause coverage problems. Besides, it is unlikely to generate sentence level paraphrases. For instance, Lin and Pantel (2001) use a single corpus collected from newspaper texts. However, the size of the corpus they used was limited and became a bottleneck for acquisition of paraphrasing as well as NLP methods. The popularization of SMT-based methods in the NLP community has led to the use of larger corpora. Pasca and Dienes (2005) shows that the limitations of a single corpus might be overcome by creating a corpus by crawling the web. However, structuring the unannotated large amount of data collected from web became another issue on usage of single monolingual corpus.

2.3.2 Monolingual Comparable Corpora

Comparable corpora consist of two different raw corpora such that there is no applied parallelism in sentence level; they rather have phrasal overlaps that are similar in terms of text type, topic and length. From comparable data, it is possible to obtain information using phrasal overlaps and/or distributional similarity measures. For example, Shinyama et al. (2002) used named entity tags, while Barzilay and Lee (2003) considered word lattices (See Section 2.5.1). However, they used a domain-dependent set of news articles, which was insufficient in terms of the amount of data required. With some preparation, comparable corpora can be used to build parallel corpora in order to increase their functionality. A sample is the Microsoft Research Paraphrase Corpus (MSPRC) which follows the same method as Barzilay and Lee (2003) and Shinyama et al. (2002) to gather data. In contrast to these, however, the MSRPC targets multiple domains to be able to create a large corpus. This is then converted to parallel corpora by applying alignment techniques.

2.3.3 Monolingual Parallel Corpora

Since the use of parallel corpora has become more common, 'text alignment' appears to be a problem in many applications. Any parallel data have to be aligned to be employed for any further process. Because alignment techniques are varied and this might significantly affect the results, the alignment techniques should be explored before using parallel texts. A few examples are noted where alignment techniques are applied to the monolingual parallel corpora. Barzilay and McKeown (2001) employ an algorithm created by Gale and Church (1991). Another alignment method is the Multiple Sequence Alignment (MSA), proposed by Barzilay and Lee (2003). MSA is a word-to-word similarity measurement that assigns a similarity score between sentence pairs. Giza++[1] (Och & Ney, 2000) is the most-widely applied alignment technique, which is used to generate SMT models from bilingual corpora. Due to the reason that it is constructed for SMT methods, it might require additional work for the paraphrasing tasks.

Barzilay and McKeown (2001) extracted paraphrases from a collection of multiple translations of classic novels written by different authors. Quirk, Brockett and Dolan (2004)

[1] http://www-i6.informatik.rwth-aachen.de/Colleagues/och/software/GIZA++.html

created a monolingual parallel paraphrase corpus from clustered daily news articles, which will be explained later.

2.3.4 Bitexts (Bilingual Parallel Corpora)

Bitexts are parallel corpora obtained by aligning sentences in documents in one language and their translations in a second language. In the context of paraphrasing, bitexts are first used by Bannard and Callison-Burch (2005). Their method takes the phrases in one language (source language), and matches them with their translation in other language (pivot language). The process is then repeated for the translated phrases in the pivot language. The translated phrases from pivot to source language and the original phrases from source language are accepted to have a paraphrase relation. In simple words, if the phrases in the pivot language correspond to two different phrases in the source language, these phrases are assumed to be paraphrases.

Even though bitexts rely on parallel texts or translation tables, they have gained favour in paraphrasing tasks via SMT applications. Using automatically generated paraphrases increase the performance of SMT systems (Bannard & Callison-Burch, 2005; Ganitkevitch et al., 2011; Madnani et al., 2007). Although there are attempts that directly target the paraphrase quality (Brockett & Kok, 2010; Callison-Burch, 2008), this remains secondary to the issue of acquisition problems.

A database of paraphrases is constructed by Ganitkevitch, Van Durme, and Callison-Burch (2013), which is one of the largest multilingual resources of paraphrases. This rich dataset is collected using Bannard and Callison-Burch's (2005) method and it is available in more than 20 languages. The dataset consists of lexical and phrasal paraphrase packages, where the smallest package represents the highest similarities, whereas the larger package includes lexical and phrasal paraphrases in a wide window of decreasing similarity scores.

2.3.5 Web as a Corpus

Most of the research on paraphrasing concludes by mentioning the problem of the lack of data sources. NLP experiments have shown that the more data is available, the more reliable the results are. Despite the vast amount of unannotated data that can be found on

the web, in practice construction of such corpora is a time consuming and costly process. Moreover, the process of corpora construction might differ for every particular NLP application.

Linguistically, a corpus is described as a ‘language sample’ that represents the language via a collection of data constructed considering the explicit linguistic criteria (Sinclair, J. cited from Duclaye et al. (2002)). Following this idea, Duclaye et al. (2002) use linguistic paraphrases and semantic derivations in order to improve question answering systems. They suggest that the web is the best ‘language sample’ to be used as a linguistic resource for their task.

This trend towards web-based corpora has emphasised the limitations of annotated corpora in the field of NLP and has led to a new interest in the use of the web as a corpus. Halevy et al. (2009) draw attention to the importance of ‘the web as a corpus’ with regard to the recent success of NLP applications that use unannotated data, since they overcome the data limitation.

STRAND is a web mining method used by Resnik and Smith (2003) to explore and extract parallel texts from the web. Their aim was to find web pages with multiple translations. The translated pages were then paired up for generating candidate pairs. Due to the lack of other language sources and the legal issues on accessing data on the Internet Archive, they extracted parallel texts only in four different languages (Arabic-English, Chinese-English, Basque-English and French-English).

Pasca and Dienes (2005) obtained paraphrases from a large monolingual corpus, extracting over a billion pages. The web was crawled using automated methods which looked for phrase-level paraphrases by checking any overlapping text fragments occurring between aligned pairs of sentences.

Large quantities of data on the web supply more informative results, but this comes with trade-offs. It is a challenging task to deal with the information on the web and considerable amount of effort is required. Also, copyright laws reduce the legal access to large amounts of data, which might be an obstacle in obtaining information from the web.

2.4 Paraphrase Identification (PI)

Paraphrase Identification is defined by Lintean and Rus (2011) as "the task of deciding whether two given text fragments have the same meaning".

In this section, we will give an overview about the type of paraphrase corpora used for paraphrase identification in the literature. A detailed description of paraphrase corpora we experimented with throughout this thesis will be provided. Section 2.4.2 will explain the methods previously used for identifying paraphrases. Then, the recent approaches for paraphrase identification will be addressed in relation to the methods based on the distributional hypothesis.

2.4.1 Corpora for Paraphrase Identification

The construction of paraphrase corpora has been an important development in research into the task of paraphrase identification. Because of the need to compare experimental results with a benchmark, there have been various attempts at constructing paraphrase identification corpora.

We experimented on three different paraphrase corpora. These are: the Microsoft Research Paraphrase Corpus (MSRPC), the Plagiarism Detection Corpus (PAN) and Twitter Paraphrase Corpus (TPC). In addition, we constructed and experimented with a new paraphrase corpus for the Turkish language. This will be discussed later, in Chapter 5.

2.4.1.1 The Microsoft Research Paraphrase Corpus

The Microsoft Research Paraphrase Corpus[2] (MSRPC) (Dolan, Quirk, & Brockett, 2004) has been used for a decade as the standard for comparison of results. As it is mentioned in Section 2.3.2, the initial dataset was comparable corpora constructed by collecting comparable newswire articles, similarly to previous approaches (Barzilay & Lee, 2003; Shinyama et al., 2002). Unlike these approaches, however, Dolan et al. (2004) use broad-domain news agencies, clustering them for a reasonable amount of time to align the pairs of sentences referring to the same event. A simple string distance supported with a heuristic observation is used to extract sentential paraphrases. The second attempt by Dolan and Brockett (2005) is continued in the development of this initial dataset, training the data

[2] http://research.microsoft.com/en-us/downloads/607d14d9-20cd-47e3-85bc-a2f65cd28042/

with Support Vector Machine classifier. Dolan and Brockett (2005) accomplished the construction of a monolingual parallel corpora: MSRPC. There are 5,802 sentence pairs in the MSRPC. Paraphrase pairs (3,900) are scored as 1 and non-paraphrase pairs (1,901) are scored as 0. The MSRPC is split into two chunks; train sets and test sets, containing 1,725 and 4,076 sentence pairs respectively. Both sets consist of randomly chosen paraphrase and non-paraphrase pairs. So far, the MSRPC has been used merely for the comparison of paraphrase identification methods. Nevertheless, the experimental data has some drawbacks, which will be revealed throughout the experiments.

2.4.1.2 Plagiarism Detection Corpus

The Plagiarism Detection Corpus (PAN) [3] is constructed by deriving aligned corresponding sentences from 41,233 plagiarised documents. It is made available by Madnani, Tetreault, and Chodorow (2012) for the use of paraphrase identification tasks, publishing the initial results of their experiment. PAN consists of 13,000 sentence pairs in total; 10,000 for the train set and 3,000 for the test set. The data contains equal numbers of paraphrase and non-paraphrase pairs in both test and train sets. It is labelled in the same way as the MSRPC: paraphrase pairs scored 1, non-paraphrase pairs 0.

2.4.1.3 Twitter Paraphrase Corpus

The Semeval-2015 Task1, "Paraphrase and Semantic Similarity in Twitter" involves predicting whether two tweets have the same meaning. Training and test data are provided in the form of a Twitter Paraphrase Corpus (TPC) (Xu, 2014). The TPC is constructed semi-randomly and annotated via Amazon Mechanical Turk by 5 annotators. It consists of around 35% paraphrases and 65% non- paraphrases. Training and development data consists of 18K tweet pairs and 1K test data. Test data is drawn from a different time period and annotated by an expert. A novel aspect of the TPC compared to other paraphrase corpora is the inclusion of topic information, which is also used during the construction process.

[3] http://bit.ly/mt-para

2.4.1.4 Judgment Criteria

Further to the previously described paraphrase corpora, a study (Rus et al., 2014) describes the basic characteristics of other paraphrase corpora constructed for paraphrase identification methods. However, one of the drawbacks of the paraphrase identification task is the lack of a fixed set of rules regarding the judgement criteria. Therefore, other corpora mostly do not fit the previously used judgement criteria for paraphrase identification task. Rus et al. (2014) discuss this issue with regard to the provided loose guidelines for the annotation of the MSRPC. They believe that a fine-grained judgement criterion should be provided for a paraphrase corpus to be used for paraphrase identification tasks.

2.4.2 Methods for Paraphrase Identification

The state-of the-art PI methods and their results can be found on Wikipedia[4]. These results are frequently updated with a short description of the applied methods and divided into two categories based on the classification technique: supervised and unsupervised. These division shows that the recent PI methods tend to use supervised approaches because they perform well as compared unsupervised approaches. This section gives an overview of the PI methods based on the applied methods rather than the classification approaches.

Previous identification methods have focused on pattern-based phrases, according to the required data. For instance, user questions and their matched answers derived from a Frequently Asked Questions (FAQs) list to obtain paraphrase questions (Tomuro & Lytinen, 2001). Another case focused on lexical 'verb ' paraphrases (Glickman & Dagan, 2003), firstly identifying the verb pairs and then extracting the pairs from a single corpus.

Inversion Transduction Grammar (ITG) works without requiring any thesaurus (Wu, 2005), and results higher than the baseline are obtained on both paraphrasing and textual entailment tasks.

Zhang and Patrick (2005) apply a small set of transformation rules on both lexical and syntactic levels using a decision tree learning method, which simplifies the text (also called 'text canonicalization'). After the transformation process, the sentence pairs that

[4] http://aclweb.org/aclwiki/index.php?title=Paraphrase_Identification_(State_of_the_art)

share commonality on surface strings are detected as candidate paraphrases. The given example follows canonicalization rules:

"Remaining shares will be held by QVC's management."

"QVC's management will hold Remaining shares." (Canonical)

Despite the fact that it has very limited features, the results are better than the baseline. One significant reason is that the paraphrase pairs in the MSRPC include high number of lexical overlaps.

Recently, a number of studies have examined machine learning methods in order to identify paraphrases. Kozarova and Montoyo (2006) measure the lexical and semantic similarity with a combination of different classifiers: k-Nearest Neighbours, Support Vector Machines, and Maximum Entropy.

WordNet is one of the resources widely used for knowledge-based paraphrase identification methods. In WordNet, lexical items are organised as sets of synonyms or "synsets". They synsets are then organised hierarchically according to lexical relations such as hyponymy and hypernymy. Similarity and relatedness of a pair of lexical items may be measured according to the hyponym/hypernym hierarchy in WordNet. The WordNet provides a *gloss (conceptual related items)* for each item, although only the items that belong to same word categories can be measured (Pedersen et al., 2004). For instance, *car* is a hypernym of *vehicle*.

A study was performed comparing corpus-based and knowledge-based measures for a semantic textual similarity task (Mihalcea, Corley, & Strapparava, 2006). A variety of WordNet[5] similarity metrics in addition to two corpus-based measures (pointwise mutual information and latent semantic analysis metrics derived from a large corpus) are experimented with on the MSRPC. Another work on semantic textual similarity focuses on only corpus-based measures utilizing semantic and string similarities (Islam & Inkpen, 2007). Their proposed method first uses three different longest common subsequence (LCS) methods for the similarities between words. Semantic similarity of a pair of

[5] http://wordnet.princeton.edu/

sentences is then measured by an advanced pointwise mutual information (PMI) algorithm (Second Order Co-occurrence PMI). Sentence similarity is computed from the joint matrix of a string similarity matrix and a semantic similarity matrix. They claim that the computational complexity is lower as compared to Mihalcea et al. (2006)'s corpus-based and knowledge-based measures, because they use only one corpus-based measure.

Fernando and Stevenson (2008) review the semantic maximal similarities between all word-to-word relations, in addition to the lexical similarities, because near-synonymy cannot be detected on a purely lexical level. Six different WordNet similarity metrics are computed. It is shown that taking into account all similarities increases the performance of the system. Malakasiotis (2009) exploits one basic method (INIT) and extends this method using WordNet (INIT-WN) and then a dependency parser (INIT-WN-DEP). First, nine different string similarity measures are applied in order to detect all possible shallow features of a pair of sentences (INIT). Searching the WordNet for synonymy relations between two sentences extend the INIT method by combining all features (INIT-WN). The third method uses a dependency parser in order to detect grammatical relations of a sentence pair (DEP) and it is combined with the previous two methods (INIT-WN-DEP).

Das and Smith (2009) use a more sophisticated approach on the syntactic and lexical levels conducted with Quasi-synchronous Grammar (QG) Formalism. This model produces a syntactic structure for two given sentences in the form of their dependency trees. The sentences are to be assumed as paraphrases when the dependency trees align closely.

Sentential paraphrases are likely to have a high degree of lexical overlap. Because of this, the dissimilarity feature has been pointed out in recent research. Qiu, Kan, and Chua (2006) realized the importance of lexical dissimilarities, as well as similarities. Wan, Dras, and Dale (2006) also consider both similarity and dissimilarity of sentence pairs from their dependency trees structure. One recent paraphrase identification method (Lintean & Rus, 2010) automatically identifies paraphrases by looking at their lexico-semantic relations for either their similarity or dissimilarity scores. Weighting the dependency score of unpaired paraphrases extends the system.

Paraphrase identification methods use similarity scores on different levels, such as word-to-word similarity, synonyms, or word matching based on a thesaurus. Lintean and Rus (2011) apply a word-to-word similarity metric on syntactic and lexico-semantic levels in order to detect the similarity between two sentences. Unlike other approaches, the similarity metric and takes account of the dissimilarity scores. These scores are obtained by computing the dependencies to decide whether the two sentences have paraphrase relation. This is based on the work of Lintean, Rus, and Graesser (2008), which examines the dependencies on MSRPC to detect paraphrase relations. However, Lintean and Rus (2011) evaluated the method of weighting the dependencies, adding a few features.

Rus, Mccarthy, Lintean, Mcnamara, and Graesser (2008) suggested a lexico-syntactic approach related to textual entailment. The method, named 'graph-subsumption', maps each input text of two corpora into a graph and performs a subsumption rule. Let's say *t* and *h* are the texts, *t* entails *h*, if, and only if, *t* subsumes *h*. A paraphrase relation is then detected by observing the *t* subsumes *h* and *h* subsumes *t*.

A number of researchers have investigated whether near state-of-the-art PI results can be obtained without use of external sources. Blacoe and Lapata (2012) use distributional methods to find the compositional meaning of phrases and sentences. They find that the performance of shallow approaches is comparable to methods that are computationally intensive or that use very large corpora. Lintean and Rus (2011) apply word unigrams and bigrams. Bigrams capture word order information, which can in turn capture syntactic similarities between two text fragments. Finch, Hwang and Sumita (2005) combines several MT metrics and uses them as features. Madnani et al. (2012) also shows that good results are obtained by combining different MT metrics. Ji and Eisenstein (2013) attain state-of-the-art results based on latent semantic analysis and a new term-weighting metric, TF-KLD.

The Twitter Paraphrase Corpus (TPC) has been released for the SemEval-2015 Task 1: Paraphrase Identification and Semantic Similarity (Xu, Callison-Burch, & Dolan, 2015). There were 18 teams participating in the main task, PI, and 13 teams participating in the semantic similarity task. The applied methods for PI range from simple similarity metrics, machine translation to a variety of neural network methods. Xu, Ritter, Callison-burch,

Dolan and Ji's (2014) approach constructs a joint word-sentence paraphrase model (MULTIP) and utilizes both word and sentence pair relations from the TPC.

Socher et al. (2011) obtained higher results than previous approaches. The Recursive Autoencoder (RAE), based on the recursive neural network model, is an unsupervised feature-learning algorithm that detects the semantic and syntactic similarities between patterns. A variable-sized similarity matrix is computed measuring the distance between pattern-based phrases. A dynamic pooling layer is then used to compute a fixed-sized similarity matrix of a sentence pair. This representation is used as an input to a classifier in order to detect a paraphrase relation between sentence pairs.

Hu, Lu, Li, and Chen (2014) propose two models of convolutional neural networks for matching sentences, and apply them to three sentence level tasks including paraphrase identification.

Another convolutional neural network approach optimized for the task of paraphrase identification is by Yin and Schütze (2015). Their approaches are based on representing sentences on multiple levels of granularity. They obtained three different results by improving their proposal with a pre-training technique and adding MT features (Madnani et al., 2012) to improve their base method.

He, Gimpel, and Lin's (2015) method first utilises another convolutional neural network approach for sentence modelling; the similarity of these obtained representations of sentences are then measured with similarity metrics without the use of any external resources. Their method outperforms the previously used methods on MSRPC as well as on Sematic Textual Similarity datasets.

A variety of classifiers has been employed for the purpose of identifying paraphrases. Kozarova and Montoyo (2006) measure lexical and semantic similarity with a combination of different classifiers: k-Nearest Neighbours, Support Vector Machines, and Maximum Entropy. SVM Classifiers remain the most applicable in recent research, whether applied on their own (Finch et al., 2005; Wan et al., 2006) or as part of combined classifiers (Kozareva & Montoyo, 2006; Lintean & Rus, 2011; Madnani et al., 2012).

Table 2-2 shows state-of-the-art results for PI methods. A distinction between the previous approaches and neural network approaches is first highlighted by He et al. (2015). The grey rows indicate the results that are obtained from neural network methods. The table is updated by adding the most recently published results.

Reference	Accuracy	F-score
Zhang and Patrick (2005)	71.9	80.7
Lintean and Rus (2010)(Opt Minipar)	72.0	80.9
Corley and Mihalcea (2005)	71.5	81.2
Yin and Schütze (2015) (without pretraining)	72.5	81.4
Qiu et al. (2006)	72.0	81.6
Blacoe and Lapata (2012)	73.0	82.3
Fernando and Stevenson (2008)	74.1	82.4
Finch et al. (2005)	75.0	82.7
Das and Smith (2009)	76.1	82.9
Malakasiotis (2009)	76.2	82.9
Wan et al. (2006)	75.6	83.0
Socher et al. (2011)	76.8	83.6
Madnani et al. (2012)	77.4	84.1
Yin and Schütze (2015) (with pretraining)	78.1	84.4
Yin and Schütze (2015) (pretraining+MT features)	78.4	84.6
He et al.(2015)	78.6	84.7
Ji and Eisenstein (2013)	80.4	85.9
BASELINE	66.5	79.9

Table 2-2: State-of-the-art paraphrase identification results on MSRPC

2.5 Other Paraphrasing Tasks

The problem of distinguishing the paraphrasing tasks is already mentioned in the Introduction. In this section, we will briefly explain the methods of paraphrase generation and extraction.

2.5.1 Paraphrase Generation (PG)

Paraphrase generation aims to generate as many paraphrases as possible from a single language expression (Androutsopoulos & Malakasiotis, 2010). Unlike identification methods, generation methods require large datasets so that they can produce multiple outputs. This suggests the usage of the web as a corpus as well as other corpora: bitexts or monolingual parallel corpora developed for SMT applications. A problem arises in the

usage of bilingual corpora due to the fact that translated texts might not correspond to exact paraphrase relation. Translators may change the meaning to a greater or lesser degree.

Statistical Machine Translation (SMT) is often associated with paraphrasing tasks, especially paraphrase generation, owing to the similarity of its approaches to the problems and also the methods, which can be successfully implemented in both tasks. It is largely accepted that the usage of SMT techniques improves paraphrasing acquisition tasks. After applying an SMT approach for extracting paraphrase pairs from an aligned monolingual corpus, Barzilay and McKeown (2001) showed that paraphrasing applications might benefit from SMT techniques. This idea was followed by the construction of a monolingual parallel paraphrase corpus, the MSRPC (Dolan et al., 2004; Quirk et al., 2004) using the current SMT tools. In addition, paraphrase generation methods, particularly SMT-based, are evaluated using the BLUE score, which is a MT benchmark for results. This is because there are no reliable comparison methods for paraphrasing tasks.

Evaluation of paraphrase generation has advanced through the idea of enlarging the data using clusters of daily news articles drawn from the web. This reveals the same daily events recounted by different authors (Quirk et al., 2004). Therefore, sentence pairs addressing the same event are extracted from clustered URLs and matched by human annotators with the assistance of a word edit distance measurement. The words in sentence pairs are aligned by an efficient word alignment technique, Giza++. Pre-processing of data takes a fair amount of time because they are not exact parallel texts; moreover, some are not full sentences, but just fragments or phrases.

Marton et al. (2009) focus on the improvement of SMT techniques considering the untranslated words that can be translated via the pivoting method. The chosen source language is English, with Chinese and Spanish as the target languages, and an unannotated monolingual corpus is used. Their system employs a distributional similarity measurement, called ‘Distributional Profiles’ (DP), for co-occurring words in the source language, assigning them a similarity score. The generated paraphrases with the highest similarity scores are added to the phrase table. This approach was extended (Marton, 2010) by adding lexical and corpus-based semantic similarity measurements. A later approach (Marton et al., 2011) takes more advantage of abundant data, focusing on antonym-related words,

which occur regularly in any corpora and can be obtained as easily as paraphrase pairs, measuring the distributional semantic distance. One noticeable point is that the selected target languages in the latter study are Chinese and Arabic, which are clearly distinct from English.

The pivoting technique performed by Zhao, Lan, Liu, and Li (2009) shows that inserting more words into the phrase table from multiple resources, rather than relying on bitexts, increases the quantity of generated paraphrases. The technique has revealed that sentential paraphrases can be obtained easily, despite the fact that the method is proposed for sentence summarisation. In addition, a combination of various methods from SMT and NLG are applied, as well as thesaurus and rule-based techniques.

Representation of paraphrases is often not easy. Brockett and Kok (2010) introduce the HTP (Hitting Time Paraphraser) Model, which is a graphical representation of paraphrase pairs. In principle, it is a pivoting-based method, but bilingual parallel corpora correspond to a graph, while a node is a phrase. Thus, if the two phrases are aligned, there appears an edge between two nodes. Another advantage of this method is that the process is not divided into two phases, so that the method can continue to search for more paraphrases, unlike in the pivoting method (Figure 2-2):

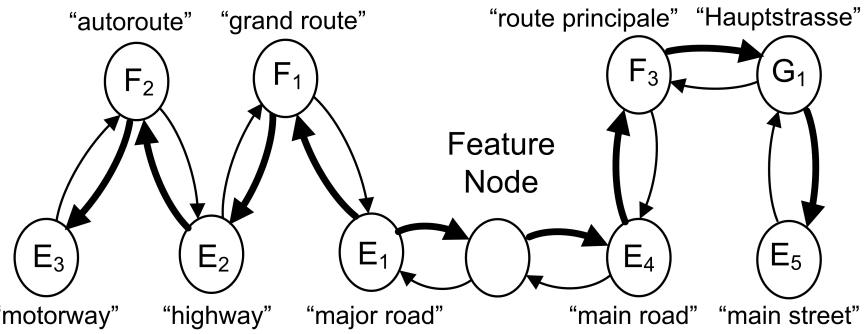

Figure 2-2 (Brockett & Kok, 2010): Graph created from English-French (E-F), English-German (E-G), and French-German (F-G) bilingual parallel corpora. Bold edges have large positive weights.

This system produces more paraphrases than the other syntactic approaches, although it used a small set of phrase tables. Brockett and Kok (2010) suggest that the system can be adjusted to any language.

2.5.2 Paraphrase Extraction (PE)

Paraphrase Extraction is the task of extracting fragments of texts with a paraphrase relation from various sources (Lintean & Rus, 2010). Extraction methods differ from both identification and generation in that there is no input. Principally, it is the matter of deriving information from a source, which contains paraphrase pairs.

A prominent approach by Barzilay and McKeown (2001) yielded new insights into the development of data used in paraphrase extraction. Multiple translations of classic novels were collected and aligned as a monolingual corpus. Translations of the same novel by different translators tend to have different form while preserving the meaning, so they tend to be paraphrases. Performing alignment techniques, each sentence in one translation is aligned to its corresponding sentence in another translation. They follow a sophisticated extraction process, which relies on similarity between two sentences. From the two aligned sentences, selected identical words are used to seed their algorithm. Identical words are identified through verb-object and noun-modifier relations. In Figure 2-3, 'Evening' is the identifier because it is the modifier of both subjects.

People said, "The Evening Noise is sounding, the sun is setting."
"The evening bell is ringing," people used to say.

Figure 2-3 (Barzilay & McKeown, 2001): Aligned sentences

In addition, lexical and syntactic features are derived for input to a classifier. A contextual classifier learns context rules from identical words, and the rules are then used to extract new paraphrase pairs.

Another method (Lin & Pantel, 2001) proposes to find 'inference rules' based on the distributional similarity hypothesis. Using unannotated monolingual corpora, the distributional hypothesis is applied to paths in a dependency tree, not to words. Therefore, the extraction of paraphrases is the result of some inference rules that identify paraphrase relations, as shown in Figure 2-4. The algorithm DIRT (Discovery of Inference Rules) automatically discovers the two paths that occur in similar contexts, assuming that the two paths are similar in meaning. In this way, they extend the distributional hypothesis:

"If two paths tend to occur in similar contexts, the meanings of the paths tend to be similar."

Convey the same meaning	X is author of Y	X wrote Y
Not exactly the same in meaning	X caused Y	Y is blamed on X

Figure 2-4 (Lin & Pantel, 2001): Inference rules

The limitations of the two approaches above (Barzilay & McKeown, 2001; Lin & Pantel, 2001) are pointed out by Ibrahim, Katz, and Lin (2003). Syntactic structures of paraphrases are derived from the dependency tree paths of aligned sentences. They estimate the frequency and context of paths, whereas Barzilay and McKeown (2001) just focus on lexical paraphrases. The method used by Ibrahim et al. (2003) is similar to DIRT. Due to the fact that the alignment technique affects the results, a different alignment function is applied, but similar results are obtained.

Realizing that the usage of comparable corpora is convenient and might be applied without alignment, Shinyama and Sekine (2003) collected data from two Japanese news agencies which refer to the same events. To extract phrase-based paraphrases, named entity tags such as names and locations are used, because these anchors increase the performance of Information Extraction systems, which is the aim of Shinyama and Sekine (2003). Their method matches the comparable sentence pairs and then the same anchors are detected in both sentences. With a dependency analyser, the two corresponding subtrees that share the same anchors are detected as paraphrases, after computing all combinations of subtrees.

Like in Shinyama et al. (2002), two comparable corpora collected from different news agencies reporting the same events are exploited by Barzilay and Lee (2003). Unlike any previous method this work focuses on sentential paraphrases. Their work also has several remarkable nuances to be noted. They clustered the sentences of the articles separately as two different corpora, in which the articles, not the sentences, explain the same events. There is no alignment technique used, nor any knowledge resources, such as a parser. They then computed the multiple-sequence alignment of two sentences to find similar patterns and to represent them as 'word lattices'. Lattices are described as compact representations of patterns (Figure 2-5). Variable words are kept in slots, so the lattice is

called “slotted”. These slotted lattices keep similar values, which are matched and identified as paraphrases.

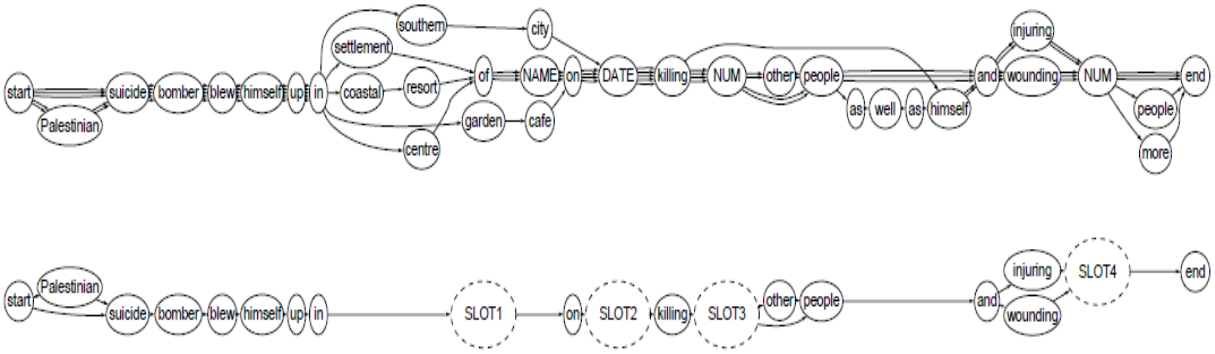

Figure 2-5 (Barzilay & Lee, 2003): Lattice and slotted lattice, derived from five sentences.

Previous extraction methods acquired paraphrases from news articles by careful structuring of the text. Unlike these, Pasca and Dienes (2005) show that the same results can be obtained with a simple approach from unstructured information, crawling the web for the sentence pairs that hold lexical overlaps after their alignment. If fragments of sentences have common word sequences, they are proposed as candidate paraphrases (Figure 2-6). Although the simplicity of the method can only handle one sense of a word, and the quality of output data are questionable, the approach is comparable to the more sophisticated approaches with regard to the usage of unstructured text from the web.

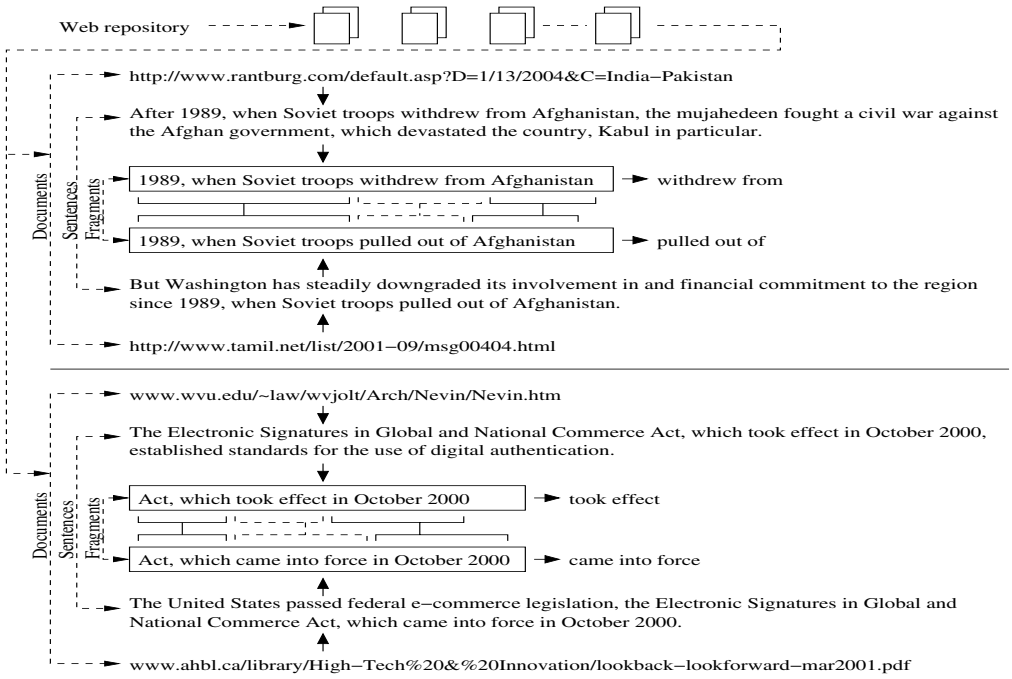

Figure 2-6 (Pasca & Dienes, 2005): Acquisition of paraphrases from the web

As mentioned earlier, the 'pivoting' technique first used to extract paraphrases by Bannard and Callison-Burch (2005) mainly relies on text alignment techniques. They create a 'phrase table' that contains aligned phrases from both source and target languages. Each phrase in the table is assigned a probability score. According to the proximity of probability scores, a similarity score is then assigned. A phrase translated from the source language to a target language phrase is found according to this similarity score. The same process repeats from target language to source language. The original phrase and the phrase obtained from second process, both in the source language, are accepted as candidate paraphrases (Figure 2-7). Furthermore, although the phrases with the highest probability are selected as paraphrases, they use a language model that can replace the other most probable phrases.

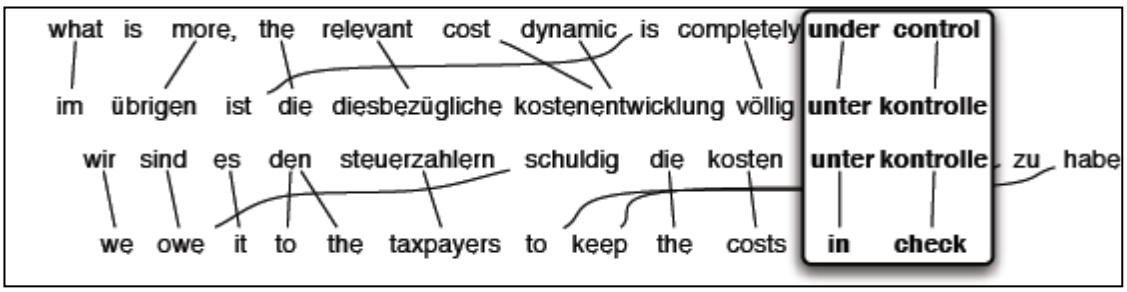

Figure 2-7 (Bannard & Callison-Burch, 2005): Paraphrase Extraction via pivoting techniques

Augmenting the data with paraphrases has advantages in dealing with out-of-vocabulary (OOV) words as shown by Marton et al (2009) using a monolingual corpus. Similarly, Callison-Burch et al (2006) found paraphrases of unknown phrases in bitexts following the same approach as Bannard and Callison-burch (2005). However, the quality of paraphrases is still questionable at this stage. Next, Cohn, Callison-Burch and Lapata's (2008) method constrains the non-constituent phrases with syntactic labels in order to increase the paraphrase quality. Manual evaluation showed that paraphrase quality was increased by 19% by applying syntactic constraints.

The extraction of paraphrases via the pivoting technique has been extended to hierarchical phrases using synchronous Context Free Grammar (CFG) (Madnani et al., 2007). Fundamentally, this research investigates whether an automatic translation system can be adjusted for evaluating reference translations, in order to decrease the need for human translations. The most efficient way is by using the automatic generation of sentential paraphrases that can be set to the reference translations. In this approach, instead of using a foreign language, English-to-English texts are used to obtain paraphrases. Therefore, synchronous CFG rules, which are designed for translation, can automatically translate English expressions to other English phrases that are supposed to be the same expressions, or paraphrases.

To avoid the data sparseness problem in a single monolingual corpus, Bhagat and Ravichandran (2008) propose a method to extract paraphrases from a large corpus, containing approximately 25 billion words. The distributional similarity hypothesis is employed to extract surface paraphrases, which in this work only correspond to words, and the syntactic paraphrases of those words appear in syntax tree. The results are not

comparable to previous work in this area and the quality of the paraphrases obtained is debatable.

Ganitkevitch et al. (2011)'s approach investigates the boundaries of bitexts adopting synchronous Context Free Grammar (CFG) rules to extract more sophisticated sentential paraphrases, in order to improve text-to-text generation tasks. They suggest that syntactic structures overcome the limitations of lexical structures. Despite the fact that it is difficult to identify sentential paraphrases, the system performs well on text-to-text generation tasks.

2.6 Summary

This literature review aimed to give an overview of computational identification of paraphrases, while recognising the widespread use of paraphrasing tasks in the field of NLP. The paraphrasing process is discussed in detail for different paraphrase tasks, differentiated as identification, generation, and extraction. A brief explanation of paraphrase generation and extraction methods is provided.

The paraphrasing concept is introduced, considering the relation of paraphrasing to the other applications. The importance of paraphrasing tasks is revealed in the field of NLP, explaining the most relevant tasks. The definition of a paraphrase is discussed, with various definitions from different authors. This makes it clear that 'paraphrasing' is more complicated than simply replacing words with their synonyms.

The importance of corpus data is emphasised. Each type of corpora is addressed, considering their limitations examined in previous research. A few advantages of annotated corpora were pointed out, some of which can be overcome by processing unannotated text, large quantities of which are available on the web.

Areas of work related to the paraphrase identification task are explained. Paraphrase identification methods are discussed with regard to the state-of-the art results. Three different paraphrase corpora for paraphrase identification task are explained in detail. The differences between these corpora are highlighted.

In the next chapter, we explore simple overlap measures as well as distributional lexical similarities for the paraphrase identification task.

Chapter 3

3 Exploiting Overlap Similarity Measures for Paraphrase Identification

3.1 Introduction

The previous chapter provided an extensive literature review of paraphrasing applications; in particular paraphrase identification methods are elaborated on with regard to the state-of-the-art results and the variety of paraphrase corpora are explained in detail.

This chapter serves as a starting point to the usage of knowledge-lean techniques. Our focus is to explore whether shallow overlap methods help to identify paraphrase pairs. We investigate the applicability of character-bigrams and lexical overlap measures, as well as further distributional word semantic similarities for the paraphrase identification task.

Experimentation on paraphrase data raises the question "how should we structure the data for the purpose of paraphrase identification?" For most NLP applications, data pre-processing is the key step that assists in discovering features of text for specified tasks. However, in paraphrasing tasks, some of these methods cause misidentification of many paraphrase sentences. The nature of paraphrases requires semantic equivalency, which can sometimes be changed even with a single lexical item. Also, the more pre-processing is applied the more useful information might be lost (See Section 3.2 for examples). One of the aims in using a knowledge-lean technique is to avoid losing information during data pre-processing. We hypothesise that knowledge-lean approaches, which avoid data pre-processing, will perform better than techniques based on pre-processed data. We explored a variety of pre-processing techniques individually, in order to determine whether these techniques are viable for paraphrase identification.

We experimented using two paraphrase corpora, MSRPC and PAN, that are structured specifically for the use of paraphrase identification tasks (explained in detail in Chapter 2 - Section 2.4.1). We investigate the different data variants in these two paraphrase corpora. Next, we explore two different approaches that use overlapping items sentence pairs. The first approach examines character and lexical overlap features, which are then used as an input to our similarity measures (Section 3.4). The second approach uses a distributional thesaurus in order to discover whether semantic information can be useful in conjunction with lexical overlap features (Section 3.5). Although we experimented using both corpora, our main experimental corpus is the MSRPC, and distributional similarity experiments are only carried out using the MSRPC. Section 3.6 covers our classification approach and evaluation measures, which are widely accepted for paraphrase identification task. We then present the results individually and in comparison to the state-of-the-art results. Lastly, we provide an analysis of our findings and conclude with a summary.

3.2 Data Pre-processing Techniques

There are plenty of pre-processing techniques that are applicable for other Natural Language Processing tasks that can also be applied to paraphrase corpora. These techniques range from shallow data normalisation methods such as tokenisation, filtering stop words, removing punctuation etc. to more structured techniques, such as part-of-speech tagging. However, in the case of paraphrasing, pre-processing techniques that remove information may be harmful. They may even decrease the accuracy of results, because there might be cases where a single token changes the meaning of a pair of sentences. For instance, the sentence pair below is marked as a negative paraphrase in MSRPC, although the words and their order are identical except for one word:

S1: NBC will probably end the season as the second most popular network behind CBS, ***although*** *it's first among the key 18-to-49-year-old demographic.*

S2: NBC will probably end the season as the second most-popular network behind CBS, ***which*** *is first among the key 18-to-49-year-old demographic.*

The sentence pair shown below is the new form of the above sentences after filtering out any other word categories except the major word ones (V, N, J, R)[6].

T1: NBC_N probably_R end_V season_N second_J most_R popular_J network_N CBS_N it_N first_R key_J 18-to-49-year-old_J demographic_J

T2: Nbc_NBC probably_R end_V season_N second_J most-popular_J network_N CBS_N first_R key_J 18-to-49-year-old_J demographic_J

The conjunction word *although* in *S1* and relative pronoun *which* in *S2* changes the meaning of the sentence. Similarities in the MSRPC depend highly on lexical overlap, which makes *T1* and *T2* highly similar. The general point is that more pre-processing is likely to remove useful information, for such cases as described, that convert a positive paraphrase pair into a negative one and vice versa.

Considering the previous discussion, we perform several techniques experimenting with both paraphrase corpora in order to find out whether pre-processing helps to identify paraphrase pairs. The shallow pre-processing techniques that we employ include tokenisation, lemmatisation, and stemming. These normalisation methods extend to abbreviation expansion, spelling correction, contraction normalisation and so on. A more advanced technique we use is part of speech tagging, where the datasets are represented with different variants of part-of-speech tagged data. This will help us provide a better judgement subsequent to morphologically examining data for disambiguation issues. Throughout the paper, we will use suffix notations that represent the type of pre-processing applied to the datasets, in addition to the name of the corpora: *MSR_Tok* shows that the tokenisation is the only method applied to the MSRPC.

The RASP Parser (Briscoe, Carroll, & Watson, 2006) is the main tool used for tokenising, morphologically analysing, and part-of-speech tagging, as well as lemmatising. Porter Stemmer is adopted from NLTK package (Bird, Loper, & Klein, 2009) for stemming.

[6] V=Verb; N=Noun; Adjective=J; Adverb=R

3.2.1.1 Data Normalisation

We have applied three different normalisation techniques: tokenisation, lemmatisation and stemming. The tokeniser detects sentence boundaries and separates each token from punctuation, such as possessive markers. The main difference between lemmatisation (as done by RASP) and Porter stemming is that the former only removes inflectional affixes and returns the lemma, whereas the latter may also remove derivational affixes and attempts to return the root (but may erroneously remove non-affixes since it does not use a lexicon of roots). Table 3-1 shows an example pair obtained from the train set of MSRPC, along with the representations after pre-processing methods are applied.

Original sentence pair	- *Amrozi accused his brother, whom he called "the witness", of deliberately distorting his evidence.* - *Referring to him as only "the witness", Amrozi accused his brother of deliberately distorting his evidence.*
MSR_Tok	- *Amrozi accused his brother , whom he called " the witness " , of deliberately distorting his evidence .* - *Referring to him as only " the witness " , Amrozi accused his brother of deliberately distorting his evidence .*
MSR_Lemma	- *Amrozus accuse his brother , whom he call " the witness " , of deliberately distort his evidence .* - *Refer to him as only " the witness " , Amrozus accuse his brother of deliberately distort his evidence .*
MSR_Stem	- *Amrozi accus hi brother , whom he call `` the wit " , of deliber distort hi evid .* - *Refer to him as onli `` the wit " , Amrozi accus hi brother of deliber distort hi evid .*

Table 3-1: A sample sentence pair from MSRPC and its representations with each data smoothing techniques applied

3.2.1.2 Part-of-speech Tagging

A further step is part-of-speech (PoS) tagging. This adds a step onto tokenised data to resolve lexical disambiguation. RASP PoS-tagger utilises a fine-grain tag set (CLAWS2 Tag Set[7]) and also analyses the text morphologically. We analysed the data with the

[7] http://ucrel.lancs.ac.uk/claws2tags.html

features of RASP Parser tool: tokenisation, sentence boundary detection, and labelling with PoS tags.

We present three different representations of PoS-tagged data. One is the main representation that includes all fine-grain tags. This form of data is notated as *MSR_PoS and PAN_PoS* for MSRPC and PAN Corpus, respectively. The other two representations examine the word categories of previously parsed data: major and closed categories. The notation *MSR_PoS_{all}* replaces all major categories with their simplified form (V, N, J, R) and includes all closed categories (including punctuation), since the usage of these categories is justifiable in terms of paraphrase applications. *MSR_PoS_{major}* is the notation for the data that only includes major categories, by leaving out all closed word categories. An example PoS-tagged sentence from MSRPC is shown in Table 3-2, along with the three different representations.

Example Sentence	*Amrozi accused his brother, whom he called "the witness", of deliberately distorting his evidence.*
MSR_PoS	*Amrozi_NP1 accuse+ed+_VVD his_APP$ brother_NN1 ,_, whom_PNQO he_PPHS1 call+ed+_VVD "_" the_AT witness_NN1 "_" ,_, of_IO deliberately_RR distort+ing+_VVG his_APP$ evidence_NN1 ._.*
MSR_PoS_{all}	*Amrozi_N accuse_V his_N brother_N ,_, whom_PNQO he_N call_V "_" the_AT witness_N "_" ,_, of_IO deliberately_R distort_V his_N evidence_N ._.*
MSR_PoS_{major}	*Amrozi_N accuse_V his_N brother_N he_N call_V witness_N deliberately_R distort_V his_N evidence_N*

Table 3-2: Representations of a PoS-tagged sentence from MSRPC

Note that the capitalisation is not changed on any of the representations of the data, unless we noted otherwise. In some cases the capitalized words cause word-sense ambiguities when they are switched to lower case. For example, the word “Apple”, a name of a brand, becomes “apple”, which is a regular noun. Because this factor carries the potential for a significant change in meaning, the datasets are morphologically analysed during the parsing process, resolving the ambiguity of words and lowercasing the generic words only.

The notations that describe the applied pre-processing for MSRPC are shown in the Table 3-3. These notations are also applied to the PAN Corpus. Simplified forms of part-of-

speech tagged MSRPC, MSR_PoS_{all} and MSR_PoS_{major}, will be used later in the distributional similarity experiment (Section 3.5).

Data Normalisation	**MSR_Tok**	Tokenized-only
	MSR_Lemma	Lemmatisation is applied on tokenised-only data
	MSR_Stem	Stemming is applied on tokenised-only data
PoS tagging	**MSR_PoS**	Part-of-speech tagged data
	MSR_PoS_{all}	Simplified form of MSR_PoS; includes all word categories
	MSR_PoS_(major}	Simplified form of MSR_PoS; includes only major categories

Table 3-3:Notations for applied data pre-processing techniques on MSRPC

3.3 Lexical and character bigram features

The most common approach for measuring the similarity between two text fragments is to look at the number of overlapping items. The literature (Chapter 2) shows that overlap features are widely used in conjunction with many NLP methods, as well as in PI methods. These features extend from characters to words to phrases and so on. The most widely used overlap features of PI methods are based on lexical and character level features.

Lexical Overlap Features: For the task of PI, highly overlapping sentence pairs might always get high scores even if they are not semantically equivalent, which means that lexical overlap might be misleading in some cases, where the context of the sentences are different (Madnani et al., 2012). As a result, the accuracy of the identified paraphrase pairs becomes an issue. However, paraphrase pairs may tend to share more lexical overlap than non-paraphrase pairs in paraphrase corpora that is used for evaluation. As indicated by Lintean and Rus (2011), paraphrase identification methods benefit from lexical unigram and bigram features. It is assumed that lexical bigrams can capture some word order information. As a result, many syntactic dependencies between two text fragments can be captured with these features. Hence, these simple lexical overlap features might at least be utilised in order to identify highly overlapping paraphrase pairs. Lexical features are obtained by looking at the word unigram features applied to each pre-processed variant of the datasets.

Character Bigram Features:

Character n-grams have shown potential for being one of the key approaches in many NLP applications. Earlier work on statistical distribution of character n-grams are used in natural language understanding and text processing (Suen, 1979) and in text categorization (Cavnar & Trenkle, 1994). More recently, it has been shown that the frequency of character n-grams can help identifying the characteristic of a language (Yang, Zhu, Apostoli, & Cao, 2007). Character n-gram tokenization also has been used as an alternative to word stemming (Mayfield & Mcnamee, 2003). N-gram measures and distances are used for measuring the word similarities by having insertion, deletion and substitution of characters. Although character n-grams are experimented in combination with variety of techniques for the STS tasks (Agirre et al., 2014, 2015, 2012, 2013), yet, it is not directly applied for the paraphrase identification tasks.

Here, we focus on simple overlapping items of adjacent character n-grams between a pair of sentences. Overlapping items are obtained from adjacent character n-grams, where n-grams are up to 4. However, we only present the results from character bigrams features, where the best results are obtained. For reference only, the results of the adjacent trigram and four-gram features, measured with Dice Coefficient, can be found in Appendix A (Table 0-2).

3.3.1 Similarity measures

Similarity measures can be varied for a number of different applications of NLP. The majority of methods for finding the similarity of a sentence pair apply a variety of string similarity measures. These measures use a few basic mathematical operations, such as intersection and union of the two items, to detect how similar the two sentences are. Each one of these addresses the absence and presence of features between sentence pairs. Most of these measures show that they can give results that are close to some sophisticated methods and can even be more reliable.

We adopted four commonly-used similarity measures: Dice Coefficient, Cosine measure, Jaccard Coefficient, and Jaccard Generalization. Table 3-4 shows the experimental measures along with their formulas and a brief explanation. These measures

provide values in the range 0–1, where value "1" shows that two sentences are identical and "0" that two sentences are not related at all.

The *Dice Coefficient* is one of the most suitable measures for paraphrasing applications that gives double weight to overlapping features between two text fragments. The overlapping items of the two sentences are counted twice in this measure. The value is then normalised by dividing it by the sum of the numbers of features representing the two sentences.

The *Jaccard Coefficient* is a straightforward way of finding similarity between two sets, dividing the size of presence features (intersection) in both sets (S_a and S_b), by the size of these two sets (union).

The next measure, *Jaccard Generalization* (Jaccard_Gen) is adapted from the Jaccard Coefficient. Sentences are represented as vectors: $S_a = (a_1, a_{2,} a_{3,} \dots, a_i)$ and $S_b = (b_1, b_{2,} b_{3,} \dots, b_i)$ with $(a_n, b_n) \geq 0$. Here, the set S_a is actually a multi-set and a_n is a count of element *n* in the multi-set S_a.

Cosine Measure is one of the most widely used measures, and calculates the cosine angle between two vectors. The lower the angle between vectors, then the similarity of sentences will be higher. We use a traditional cosine similarity measure commonly used in information retrieval for document-term similarity. Here we take into account the number and frequency of words based on word occurrence. The presence of a feature indicates the frequency of occurrence of a word in a sentence; 0 in the case of feature absence.

The lexical and character bigrams features of each data variant are computed with the measures defined below.

Measures	Formulas	Explanation
Dice Coefficient	$\frac{\mathbf{2}\times \lvert S_a \cap S_b \rvert}{\lvert S_a \rvert + \lvert S_b \rvert}$	The size of the intersection is double weighted, and this is divided by the sum of the numbers of features representing the two sentences, S_a and S_b.
Jaccard Coefficient	$\frac{\lvert S_a \cap S_b \rvert}{\lvert S_a \cup S_b \rvert}$	The size of the intersection is divided by the size of the union of the sets of two sentences, S_a and S_b.
Jaccard Generalization	$\frac{\sum_n min(a_n, b_n)}{\sum_n max(a_n, b_n)}$	Sentences, S_a and S_b, are represented as a vectors (multi-sets).
Cosine Measure	$\frac{\overrightarrow{S_a}\,\overrightarrow{S_b}}{\lVert S_a \rVert \, \lVert S_b \rVert}$	The cosine angle between two vectors. Sentences are mapped into d dimension vectors, where the d is the set of words in a pair of sentences and the value of a feature is the number of occurrences in a given sentence; 0 otherwise.

Table 3-4: Experimented similarity measures, their formulas and brief explanations

3.4 Distributional Thesauri for Paraphrase Identification: Byblo Thesaurus

Previously, we have examined several overlap measures by looking at the absence and presence of the items in a sentence pair, where the semantic similarity information is not taken into account. In this section, we propose more advanced knowledge-lean approaches capturing lexico-semantic information, where semantic relations of lexical items are derived from a distributional thesaurus. A distributional thesaurus is a thesaurus where every pair of lexical items is associated with a similarity score. The scores are computed according to a measure of distributional similarity, based on the distributional hypothesis (Firth, 1957) which implies that words that tend to appear in similar linguistic contexts will tend to have similar meanings. The notion of linguistic context here is not fixed and might be modelled in a variety of different ways. For example, two words might be considered to inhabit the same context if they appear in the same document or the same sentence or if they stand in the same grammatical relationship to some other word (e.g. both occur as subject of a particular verb or modifier of a given noun). This contrasts with a manually

created resource such as WordNet, where lexical items are organised considering their lexico-semantic relations.

Mihalcea, Corley, and Strapparava's (2006) approach differentiates the corpus-based similarities from knowledge-based similarities (WordNet) for the task of semantic similarity, which is evaluated using the MSRPC. They use two corpus-based measures, Pointwise Mutual Information and Latent Semantic Analysis, used to collect word semantic similarities from the web and the British National Corpus, respectively. Mihalcea, Corley, and Strapparava (2006) consider only maximum word-to-word similarities among sentences, ignoring the other word similarities. Islam and Inkpen's (2007) corpus-based method utilises 3 different longest common subsequence string similarity measures. They indicate the usage of WordNet that increases the run time and the complexity of methods in comparison to Mihalcea et al. (2006)'s knowledge-based methods.

WordNet similarity scores are widely used, due to its robustness and reliability. Fernando and Stevenson (2008) construct a symmetric word similarity matrix for six different similarity metrics from WordNet by deriving all word-to-word similarities, in contrast to the approach of Mihalcea et al. (2006), which considers only maximum word-to-word similarities among sentences by having the score of the most similar word and ignoring the rest of the word similarities, in order to obtain a paraphrase score for each sentence pair. They then apply a simple metric, more like a cosine similarity, to acquire the results.

The work on the usage of corpus-based similarities has established a motivation for using a distributional thesaurus for the task of PI. Our objective is to find the extent of the applicability of distributional word similarities for the identification of paraphrase pairs from a distributional thesaurus, rather than from WordNet. Fernando and Stevenson's (2008) approach can be adapted in a straightforward way to distributional similarity scores. Our approach uses semantic information from a distributional thesaurus by computing a similarity score that operates on word-to-word similarity scores, to compute the similarity score of a sentence pair. The way the word similarity scores are used is adjusted from Fernando and Stevenson's (2008) method.

The next section describes the distributional thesaurus that is used in our experiments. Later, the usage of word similarity scores, and the implementation details of this method, will be described.

3.4.1 Byblo Thesaurus

Byblo is a software tool used to create a distributional thesaurus automatically by constructing a statistical model from unlabelled text, unlike WordNet or any other manually created thesaurus. Byblo Thesaurus[8] is constructed from the Wikipedia mid-2011 dump files in order to find all pair-wise similarities of terms. A collection of individually tagged files (nouns/verbs/adjectives/adverbs) is obtained with the implementation of Byblo[9] (Morgan, 2011).

The corpus is tokenised, lowercased and then a list of all unique items (entries) is obtained. Features of each entry are selected based on the frequency of occurrences of all other entries within a window size of +/-1. Next, the occurrences of each feature with each unique entry are counted. A multi-set of all the features for each entry is converted into a vector. The similarity between each entry is measured by computing the cosine angle between their feature vectors.

In Byblo Thesaurus, each entry word (root word) is aligned with its top 100 neighbours, where a similarity score follows every neighbour in descending order. Although there are only 100 most similar neighbours included for each entry, the similarities are symmetric so that the similarity of the two entries, *a* and *b,* is same as similarity of *b* and *a*. Byblo Thesaurus has 151,006 root words in total. Table 3-5 shows the statistical figures for each of the major word categories in Byblo Thesaurus.

[8] The collection was kindly provided by Julie Weeds (juliewe@sussex.ac.uk)
[9] https://github.com/MLCL/Byblo

BYBLO	
Nouns	124,343
Adjectives	18,572
Verbs	6,132
Adverbs	1,959
Total	151,006

Table 3-5: The number of major category words and total words in Byblo Thesaurus

In general, a distributional thesaurus gives content with the results, so that we can see many synonyms or/and near-synonyms together with a score that shows how close they are to the root word. However, a distributional thesaurus is constructed based on the distributional hypothesis, that is, the words that occur in similar contexts are likely to be closer in meaning. The problem with distributional similarity scores is that words with related meanings may in fact have high similarity scores, but the meaning relationships may not be synonymy but rather antonymy, hypo/hypernymy or co-hyponymy. For instance, although antonym words, "hot" and "cold" are opposite in meaning, they tend to appear in similar contexts and will therefore be scored as highly similar.

3.4.2 Distributional Word Similarity for Paraphrase Identification

We combined two approaches: following Mihalcea et al.'s (2006) corpus-based application, we obtained the distributional word semantic similarities from a distributional thesaurus considering the maximum similarities, and like Fernando and Stevenson (2008) we took into account all word-to-word similarities in addition to maximum similarities.

Word similarities from unlabelled data result in ambiguities. For example, the word "book" can be a verb that means "reserve", but also is used as a noun that means "reading material"; they are semantically unrelated. Use of word similarity scores from unlabelled data causes incorrectly identified paraphrase pairs, whereas PoS-tagged data will have both verb and noun forms of the word "book", so that the similarities can be used correctly.

3.4.2.1 Data Preparation

For this experiment we use the two sub representations of MSR_PoS: MSR_PoS_{all} and MSR_PoS_{major}. A small subset of the thesaurus exclusive to MSRPC is constructed from Byblo Thesaurus. Since our thesaurus consists of items that are labelled only with major word categories, we use MSR_PoS_{major} data to acquire all unique items in order to create a smaller thesaurus that will be used for our experiment. Since Byblo does not

have any capitalised words, we lowercased all capital letters in the dataset, to take advantage of all the items.

In total, there are 13,846 tokens found in the MSRPC. We look for each token in Byblo Thesaurus and also in WordNet, just to compare the quantity of tokens that can be found using a structured or unstructured thesaurus. The number of tokens found in Byblo Thesaurus is 11,560, whereas WordNet contains 9,275 tokens in total. Table 3-6 shows the number of unique tokens individually and in total.

Word Categories	MSRPC Tokens	Tokens found in WordNet	Tokens found in Byblo
Nouns	9,420	5,677	7,808
Adjectives	2.003	1,349	1,521
Verbs	1,934	1,811	1,800
Adverbs	489	438	431
Total	**13,846**	**9,275**	**11,560**

Table 3-6:Number of unique tokens in MSRPC; found in WordNet and in Byblo

Considering the individual major category tags, WordNet has better coverage than the Byblo Thesaurus of adverbs and verbs. However, WordNet cannot find a high proportion of the nouns and a fair amount of the adjectives. This is because WordNet, which only contains isolated words, does not include proper nouns, whereas Byblo Thesaurus contains more nouns, such as proper nouns, compound nouns etc.

One problem with similarities is that they do not show any relation between words in different word categories. For instance, the word “decision” (noun) and the word “decide” (verb) do not have any similarity score, because the similarity of two words is defined only if they belong to same word categories.

3.4.2.2 Implementation Details

The next step is to derive the similarities of each word pair in paraphrase sentences. Pairs of sentences (S_1, S_2) are constituted by words; w_m^1 where *m* is the number of unique words in S_1, and, w_n^2 where *n* is the number of unique words in S_2.

$$S_1 = \{w_1^1, w_2^1, \dots, w_{m-1}^1, w_m^1\} \qquad S_2 = \{w_1^2, w_2^2, \dots, w_{n-1}^2, w_n^2\}$$

For each w^1in S_1, we obtained the word-to-word similarity score to each w^2in S_2. The obtained set includes subsets that are specific to each w^1 in S_1, where *subset_1*

includes all the word similarities between (w_1^1, w^2); *subset_2* for (w_2^1, w^2)…*subset_m* for (w_m^1, w^2). Conversely, the similarity scores between each w^2in S_2 and each w^1in S_1 are same for the symmetric similarity matrix in Fernando and Stevenson (2008) experiment. An example case shown in Figure 3-1 is the similarity of the two sentences: "The dog sat on the mat" and "The mutt sat in the rug".

	dog	mat	mutt	rug	sat
dog	1	0	0.8	0	0
mat	0	1	0	0.9	0
mutt	0.8	0	1	0	0
rug	0	0.9	0	1	0
sat	0	0	0	0	1

Figure 3-1: (Fernando & Stevenson, 2008) The word-to-word similarities between two sentences

However, in our case, Byblo Thesaurus might not show all similarities, due to its limitation of 100. For instance, the root word *"powell"* appears to have a similarity score with the word *"michael"*, but the root word *"michael"* does not includes the *"powell"* in its top 100 neighbours. To overcome this problem, we ran a script to obtain complete symmetry by extending the number of neighbour words, which is acquired, adding any reversed word similarities that are absent.

Fernando and Stevenson (2008) take into account all similarities between each pair of words in sentence pairs by applying the similarity matrix. They calculate the similarity of sentence pairs using the formula below (Equation 3.1), where *a* and *b* sentences of pairs are presented as binary vectors according to the presence or absence of a word. *W* is defined as the semantic similarity matrix containing all word-to-word similarities.

$$sim_{(\vec{a},\vec{b})} = \frac{\vec{a}W\vec{b}}{|\vec{a}||\vec{b}|} \qquad (3.1)$$

Equation 3.2 takes into account all word-to-word distributional similarity scores by summing the set of similarities obtained from each sentence pair.

$$sum_{sim(S_1,S_2)} = \sum_{\substack{0 \le i \le m \\ 0<j<n}} sum\_similarity(w_i^1, w_j^2) \qquad (3.2)$$

In addition, we also consider the maximum similarities with reference to Mihalcea et al.'s (2006) approach. Considering maximum similarities has the advantage of taking the

similarities into account of all non-overlapping items. Equation 3.3 considers only the sum of maximum similarities between every word pair, ignoring the other similarity scores.

$$max_{sim(S_1,S_2)} = \sum_{\substack{0\leq i\leq m \\ 0<j<n}} maximum\_similarity(w_i^1, w_j^2) \qquad (3.3)$$

Maximum word-to-word similarity (*max*) is plausible because it can distinguish between a case where a word in one sentence is very similar (or identical) to one in another, and a case where a word is vaguely related to lots of words in another sentence. Summing may not distinguish these cases very well; we hypothesised that max may do better than summing. Hence, we experimented with both summing and taking maximum word-to-word similarities.

The results acquired from the above equations are balanced using 3 different normalisation techniques. The first one is obtained by computing the length of the square root of the two sentences length (Equation 3.4). This method is named as "Sum1" if the numerator is Equation 3.2, or "Max1" if the numerator is Equation 3.3. The next one is "Sum2", the product of two sentences' length (Equation 3.5). The last one is "Sum3", the size of the unique elements of the two sentences that are consisted of word tokens (Equation 3.6).

$$Sum1_{sim(S_1,S_2)} = \frac{\sum_{\substack{0\leq i\leq m \\ 0<j<n}} Sum(w_i^1, w_j^2)}{\sqrt{length(S_1)}\sqrt{length(S_2)}} \qquad (3.4)$$

$$Sum2_{sim(S_1,S_2)} = \frac{\sum_{\substack{0\leq i\leq m \\ 0<j<n}} Sum(w_i^1, w_j^2)}{length(S_1)length(S_2)} \qquad (3.5)$$

$$Sum3_{sim(S_1,S_2)} = \frac{\sum_{\substack{0\leq i\leq m \\ 0<j<n}} Sum(w_i^1, w_j^2)}{|tokens(S_1)\cup(tokens(S_2)|} \qquad (3.6)$$

The same notation applies to the sum of maximum similarities: Max2 or Max3, if the numerator is Equation 3.5 or Equation 3.6, respectively. A similarity score, ranging between 0 and 1, is then obtained for each sentence pair.

3.5 Classification Methods and Evaluation Measures

3.5.1 Classification

Classification is the task of assigning items of data to predefined data classes. Classifiers may be developed by fitting a model to pre-classified (labelled) training data (supervised training). The resulting models can then be applied to unlabelled data to predict class labels for data items.

Our classification methods are based on tuning a threshold. The two paraphrase corpora are already split into two chunks: train set and test set. Our binary classifier finds the best possible statistical model that fits the training set. This model is then applied to the unseen data (test set) for prediction. The methodology of our classifier follows the steps:

1. Similarities of paraphrase pairs are calculated for the train set by applying the specified method.
2. The results previously obtained for the train set are observed according to known input labels of paraphrase pairs.
3. A threshold value is chosen where the highest accuracy was obtained on the training set.
4. Similarities of paraphrase pairs are calculated for the test set by applying the specified method.
5. The threshold obtained from the train set is merely applied for processing the test set for each system.
6. Results are evaluated against the input labels for the tuned threshold on the test set for prediction.

The tuned threshold results on the test set are then accepted as the actual results which show how well the classifier predicts whether a sentence pair is paraphrase or not.

3.5.2 Evaluation Measures

The applicable evaluation measures, which are also standard for most NLP applications, are *accuracy*, *precision*, *recall* and, *F-score*, by which all experimental results on the MSRPC are interpreted by researchers. These measures can be acquired from classified outcomes of correct predictions (True Positive=TP and True Negative=TN) and incorrect predictions

(False Positive=FP and False Negative=FN). Equation 3.6 shows the formulas of these measures.

$$Precison = \frac{TP}{TP+FP} \quad Recall = \frac{TP}{TP+FN} \quad Accuracy = \frac{TP+TN}{TP+TN+FP+FN} \qquad (3.6)$$

We evaluated our results based on these measures. Precision and recall are known to best find how well the classifier performs. Overall performance can be interpreted from F-score, the harmonic mean of the precision and recall (Equation 3.7), which is contingent upon the accuracy of the system.

$$F\text{-}score = \frac{2xPrecisionxRecall}{Precision+Recall} \qquad (3.7)$$

3.5.3 Baselines

There are simple approaches to setting a baseline of any given dataset. A widely used approach is labelling all instances as positive. Randomly labelling the instances of a dataset as true or false, with equal probability is another approach (Corley & Mihalcea, 2005).

For the MSRPC, as well as the computation of a random baseline, a vector-based similarity measure is performed in supervised and unsupervised manners (Mihalcea et al., 2006). Among researchers, the widely accepted baseline is the supervised vector-based similarity measure, which is a cosine similarity measure with *tf-idf* weighting. The baseline for the PAN Corpus has been set by combining different MT metrics (Madnani et al., 2012). Base metrics are defined as BLEU, NIST and TER. Table 3-7 present the baselines from both datasets that were used in our experiments.

The distribution of datasets with regard to the number of positive and negative paraphrases is quite different, and so are the baselines.

Data	Baselines	Acc.	Pre.	Rec.	F-sc.
MSRPC	Vector-based (supervised)	66.5	66.5	100	79.9
PAN	Base Metrics	88.6	-	-	87.8

Table 3-7: Baseline results for MSRPC and PAN

3.6 Results and Analysis

Firstly, we observe the results from the methods that use overlap measures on different variants of pre-processed corpora. The results are obtained from each paraphrase corpus by tuning the threshold for each data variant and overlap measure separately. Next, distributional word similarity results are presented. These results are analysed and compared to the results of corpus-based methods (Islam & Inkpen, 2007; Mihalcea et al., 2006).

3.6.1 Overlap Measures Results

3.6.1.1 Results for MSRPC

The Table 3-8 shows the results from overlap measures of lexical and character level features on tokenised, lemmatised and stemmed MSRPC data. The best performing measure that utilises lexical features is the Dice Coefficient on MSR_Lemma. In general, lexical features perform well with similarity measures. The Cosine measure does not perform as well as similarity measures on each data variant.

As for the character bigram features, the highest accuracy and F-score was obtained with the Jaccard Generalization measure on tokenized MSRPC. Although accuracy results are slightly lower with Cosine on each data variant, F-score results are quite competitive.

		Dice		Cosine		Jaccard		Jaccard Gen	
Lexical features	**Data Variants**	**Acc.**	**F-sc.**	**Acc.**	**F-sc.**	**Acc.**	**F-sc.**	**Acc.**	**F-sc.**
	MSR_Tok	73.22	81.20	71.65	81.23	73.16	81.25	73.10	80.86
	MSR_Lemma	73.51	81.67	71.83	79.60	73.22	81.17	73.16	80.99
	MSR_Stem	73.16	81.13	72.12	80.26	73.33	81.30	72.41	80.70
		Dice		**Cosine**		**Jaccard**		**Jaccard Gen**	
Character bigram features	**Data Variants**	**Acc.**	**F-sc.**	**Acc.**	**F-sc.**	**Acc.**	**F-sc.**	**Acc.**	**F-sc.**
	MSR_Tok	73.10	81.14	72.06	80.18	73.16	81.09	74.09	82.31
	MSR_Lemma	73.22	81.14	72.29	80.96	73.22	81.03	73.91	81.57
	MSR_Stem	73.74	81.93	72.41	81.73	73.28	81.42	73.39	81.39

Table 3-8: The results from overlap measures that use lexical and character bigrams features on different data variants of MSRPC

It seems that the character bigrams results tend to be higher than the lexical results. With reference to overall results, similarity measures (Dice, Jaccard and Jaccard_Gen) perform better on any data variant, whereas Cosine results tend to decrease with both lexical and character bigram features.

As far as we know, previous research on paraphrase recognition directly applies PoS tagging and/or stemming where tokenisation is already included, but tokenisation alone has not been considered. Our results show that tokenisation alone can be very powerful.

	Dice		Cosine		Jaccard		Jaccard Gen	
Data Variants	**Acc.**	**F-sc.**	**Acc.**	**F-sc.**	**Acc.**	**F-sc.**	**Acc.**	**F-sc.**
MSR_PoS	72.06	79.93	72.06	81.30	72.23	80.86	72.70	81.11
MSR_PoS_{all}	73.39	81.30	71.65	80.46	73.39	81.36	72.64	80.78
MSR_PoS1_(major}	69.62	78.66	70.61	79.28	69.57	78.39	70.72	78.68

Table 3-9: The results from overlap measures using lexical features on part-of-speech tagged MSRPC

Table 3-9 demonstrates the PoS-tagged data results. The overall highest accuracy is obtained on MSR_PoS_{all} applying Jaccard Coefficient. The overall results show that the experimentation on the dataset that includes all word categories increases the quantity of identified paraphrase pairs. MSR_PoS_{major} tend to give the lowest results, consequently leaving out closed category tags is not really useful. The above PoS-tagged

data results show that filtering punctuation, stop words etc. is not really helpful in terms of paraphrasing methods. Therefore, we can argue that PoS-tagged data can be utilised for paraphrase identification, and can even improve the results by employing reduction on word categories, unless we are only considering the major word categories.

Table 3-10 represents the state-of-the-art results. Our highest score is 74.1 and 82.3 accuracy and F-score, respectively. This result is obtained from Jaccard Generalization using character bigrams features of tokenized MSRPC. It can be seen that results based on simple overlap can be competitive when compared with the relatively sophisticated methods represented in Table 3-10.

Method	Acc.	F-sc.
Madnani et al (2012)	77.4	84.1
Socher et al. (2011)	76.8	83.6
Malakasiotis (2009)	76.2	82.9
Das and Smith (2009)	76.1	82.9
Wan et al. (2006)	75.6	83.0
Finch et al. (2005)	75.0	82.7
Fernando and Stevenson (2008)	74.1	82.4
Lintean and Rus (2010)	72.0	80.9
Qiu et al. (2006)	72.0	81.6
Zhang and Patrick (2005)	71.9	80.7
Corley and Mihalcea (2005)	71.5	81.2
BASELINE	66.5	79.9

Table 3-10: State-of-the-art results from the MSRPC

3.6.1.2 Results for PAN

Table 3-11 shows the results of similarity measures used to experiment with three different representations of the PAN Corpus at lexical and character bigram level.

	Data Variants	Dice		Cosine		Jaccard		Jaccard Gen	
Lexical features		**Acc.**	**F-sc.**	**Acc.**	**F-sc.**	**Acc.**	**F-sc.**	**Acc.**	**F-sc.**
	PAN_Tok	88.87	88.55	75.50	71.91	88.87	88.52	87.60	86.62
	PAN_Lemma	88.67	88.39	76.37	72.38	89.07	88.72	87.47	86.68
	PAN_Stem	88.07	88.77	75.93	72.12	89.07	88.70	88.23	87.47
Character bigram features	**Data Variants**	**Dice**		**Cosine**		**Jaccard**		**Jaccard Gen**	
		Acc.	**F-sc.**	**Acc.**	**F-sc.**	**Acc.**	**F-sc.**	**Acc.**	**F-sc.**
	PAN_Tok	83.53	81.48	73.07	67.71	83.67	81.55	85.00	83.32
	PAN_Lemma	83.73	81.67	73.20	67.94	83.87	81.75	84.57	83.28
	PAN_Stem	84.47	83.14	72.00	67.34	83.00	80.59	83.90	82.54

Table 3-11: Results from overlap measures that use lexical and character bigram features on different data variants of the PAN Corpus

For lexical level features, the highest F-score is obtained on PAN_stem with the Dice measure, with the accuracy quite low compared to the Jaccard results on PAN_Lemma. The similarity measures at lexical level perform well on each dataset, but Cosine measures remarkably low in terms of both accuracy and F-score.

For character bigram features, PAN_Tok with Jaccard Generalization holds the highest accuracy and F-score. Although similarity measures work well with character bigram features, Cosine results are decreased remarkably on each dataset.

Overall, the table shows that the Dice, Jaccard and Jaccard Generalization results are mostly steady, whereas the Cosine Measure, based on vectorial representation of words, gives remarkably lower results. As compared to word-level measures, a notable decrease occurs in all results, although Cosine results, because they are low overall, are less affected than other measures. It seems that due to high lexical overlapping items, cosine tends to accept most negative pairs as positive. For instance, the error rate for PAN_lemma using character bigrams features is 0.26 (false positive rate is 0,10) with cosine, whereas dice has the value of 0.16 error rate (false positive rate is 0,05) with the same set of features.

Table 3-12 presents the results from 3 representations of PoS-tagged PAN Corpus. The Jaccard Coefficient is the best-performing measure on PAN_PoS_{all} and Dice Coefficient results are also very competitive for each dataset, outperforming the Jaccard Coefficient on PAN_PoS2_{major} with a slight increase in accuracy and F-score. The

most noticeable change is the performance of Cosine, which increases notably on PAN_PoS_{major}, which consists of only major word categories.

	Dice		**Cosine**		**Jaccard**		**Jaccard Gen**	
Data Variants	**Acc.**	**F-sc.**	**Acc.**	**F-sc.**	**Acc.**	**F-sc.**	**Acc.**	**F-sc.**
PAN_PoS2	89.47	89.18	75.70	72.29	89.53	89.24	88.40	87.62
PAN_PoS_{all}	89.77	89.41	76.93	73.91	89.80	89.61	88.37	87.95
PAN_PoS_{major}	89.63	89.43	87.13	86.30	89.60	89.19	89.13	88.74

Table 3-12: The result from overlap measures on part-of-speech tagged PAN Corpus

Table 3-13 presents the state-of-the-art-results for the PAN Corpus. The left-hand side of the table shows the individual performance of eight Machine Translation (MT) metrics. The right-hand side of the table shows the performance of a combination of these eight metrics as features. In spite of the fact that individually their metrics do not always perform well compared to our results, the combination of these metrics outperforms them on PAN. Our highest results are 89.80 and 89.61, accuracy and F-score, respectively. These results are obtained on PAN_PoS_{all} by applying the Jaccard Coefficient and they are still lower than the results that are obtained from combined machine translation metrics. One of the reasons is that their TERp metric performs better, because it has the feature of detecting phrasal paraphrases, whereas our measures operate only at character and lexical level. However, our simple overlap methods perform better than individually-applied MT metrics.

Metric	**Acc.**	**F-sc.**	**Features**	**Acc.**	**F-sc.**
MAXSIM	84.7	83.4	Base Metrics	88.6	87.8
BADGER	88.5	87.9	+TERp	91.5	91.2
SEPIA	87.7	86.8	+METEOR	92.0	91.8
TER	85.7	83.8	+BADGER	92.3	92.1
BLEU (1-4)	88.9	87.1	Combined 8 MT metrics	92.3	92.1
NIST (1-5)	88.2	87.3			
METEOR	89.5	88.9			
TERp	91.2	90.9			

Table 3-13: Madnani et al. (2012) state-of-the-art results from PAN Corpus

Performance on the PAN is generally higher than that on the MSRPC. As we can see, there is a notable difference between MSRPC and PAN results: MSRPC results tend to

increase when performing character bigrams, in contrast to the PAN Corpus, whose results are considerably lower for character bigram measures. This issue is pointed out by Madnani et al. (2012), who believe the negative instances in the PAN Corpus are easier to predict than the MSRPC. For instance, we examined the character bigram features of both lemmatised datasets and applied the Dice coefficient. In the PAN corpus, 1,164 pairs, approx. one third of the test set, scored higher than 0.7 (false positive rate is 0,05), whereas in MSRPC, the number of sentence pairs that scored above 0.7 (false positive rate is 0,54) is 1,258, more than two-thirds of test set.

As a result, it can be seen that data smoothing techniques and PoS-tagged information might help to identify more paraphrase pairs. However, the performance of these varied methods might change with reference to different datasets and/or similarity measures performed. For the MSRPC, data smoothing techniques performed well, whereas the best results for the PAN Corpus were obtained using PoS-tagged information.

3.6.2 Distributional word similarity results on MSRPC

Table 3-14 shows the results of the distributional word similarity experiment.

	Sum1		**Sum2**		**Sum3**	
Data Variants	**Acc.**	**F-sc.**	**Acc.**	**F-sc.**	**Acc.**	**F-sc.**
MSR-PoS_{all}	68.87	79.80	68.64	80.23	70.26	80.29
MSR-PoS_{major}	67.83	79.27	66.90	79.69	69.57	78.96
	Max1		**Max2**		**Max3**	
Data Variants	**Acc.**	**F-sc.**	**Acc.**	**F-sc.**	**Acc.**	**F-sc.**
MSR-PoS_{all}	68.93	79.77	68.23	79.82	70.09	80.09
MSR-PoS_{major}	68.29	78.01	66.43	79.78	69.45	79.41

Table 3-14: Distributional word similarity results on MSRPC

Among the Sum results, Sum3 is the best performing method on MSR_PoS_{all}, although the lowest F-score is obtained with Sum3 on MSR_PoS_{major}. The highest results of the Max method are obtained with Max3 on MSR_PoS_{all}.

In terms of the datasets, the results of MSR_PoS_{all} are higher than that of MSR_PoS_{major} for both sum and maximum word similarity experiments. The highest accuracy overall was obtained from *Sum3* on MSR_PoS_{all} in overall.

Previous corpus-based results are presented in Table 3-15 in comparison to our best results. The highest accuracy is acquired on MSR_PoS_{all} by Sum3 method, where the sum of word similarity scores is normalised by dividing it by the size of the set of sentences. Normalising the similarity scores using different combinations of the sentences' length changes the system performance slightly. The highest results are obtained with Sum3, where the sum of distributional similarities is normalised by the size of set of lexical items in a sentence pair. It might be the fact that the scores obtained from Sum1 and Sum2 are not easy to separate with a simple threshold because of the ratio of numerator and denominator. The results are low compared to the overlap measures results, but can be competitive in comparison with the results of corpus-based measures.

Method	**Acc.**	**F-sc.**
MSR_PoS_{all}-Sum3	70.3	80.3
MSR_PoS_{all}-Sum2	68.6	80.2
Method	**Acc.**	**F-sc.**
STS (Islam & Inkpen, 2007)	72.6	81.3
PMI-IR (Mihalcea et al., 2006)	69.9	81.0
LSA (Mihalcea et al., 2006)	68.4	80.5
BASELINE	66.5	79.9

Table 3-15:Comparison with the corpus-based results of Mihalcea et al. (2006) and Islam and Inkpen (2007)

The Pearson correlation coefficients are between similarity scores computed by the various methods: The cosine measure of MSR_Lemma (character and lexical level) and sum3 measure of MSR_PoS_{all}. The correlation results shown in Table 3-16 are based on test data. There is a high correlation between the character bigrams and lexical features. Although the correlation between MSR_Lemma and MSR_PoS_{all} is lower, distributional word similarities are somewhat more correlated with character bigrams as compared to lexical features.

The improvement of similarity measures is tested on MSR_Lemma (character and lexical features) and sum3 using a two-tailed t-test, in order to determine whether the

similarities are obtained by chance. The obtained scores show that the results are statistically significant at the 0.05[10] confidence level (*p*-value <0.05) (Table 3-16).

Data	MSR_Lemma (Cosine measure)		MSR_PoS_{all}
Features	**Lexical**	**Character bigrams**	**Sum3**
Lexical	1.0	0.866	0.567
Character bigrams	*t*-value:-9.74912 *p*-value <.00001	1.0	0.666
Sum3	*t*-value-28.87251. *p*-value <.00001	*t*-value -41.94145. *p*-value <.00001	1.0

Table 3-16: Pearson Correlation and two-tailed t-test results

3.7 Discussion and Overall Summary

In this chapter, we explored overlap measures that utilise character bigrams and lexical features on different variants of pre-processed corpora. Usage of distributional semantic similarity information was then performed by means of a distributional thesaurus, instead of a structured thesaurus such as WordNet. Two paraphrase corpora were experimented with in this chapter: the MSRPC was the main corpus of our experiments and the PAN Corpus was used only for experimenting with overlap methods.

A variety of normalisation techniques were performed individually on two paraphrase corpora. The usage of a more advanced technique, part-of-speech tagging, decreases the overall performance slightly with closed category words, whereas there is a notable decrease in performance for major category tags. That means less pre-processing might give results comparable to the methods that apply many pre-processing techniques. However, their performance is varied and depends on the type of dataset used.

One significant finding is observed on the PAN Corpus using the Cosine measure. The results of the Cosine measure, with both lexical and character features are notably lower compared to other similarity measures.

Overlap measures are simple, yet they are the basis for identification of paraphrases so that they can be more reliable and underlie the majority of sophisticated techniques. In

[10] http://www.socscistatistics.com/tests/Default.aspx

spite of this, usage of word semantic similarity scores show that the obtained results are competitive compared to the previously applied corpus-based measures on the MSRPC. However, distributional semantic similarity methods perform worse than overlap methods overall. One reason might be the data coverage on the MSRPC, which is approximately 83% using Byblo Thesaurus, meaning that the 17% of the tokens cannot be utilised for word similarities.

This chapter set out to explore the basis of overlap measures and distributional similarity measures with a simple classifier based on a tuned threshold, for paraphrase identification. The next chapter investigates the usage of simple overlap features, employing more advanced classifiers. Chapter 6 also explores distributional similarity theory, using a different method and a very large corpus.

The next chapter focuses on a colloquial dataset and we explore the usage of simple overlap features with SVM classifiers.

Chapter 4

4 Paraphrase Identification using Simple Overlap Features and SVMs

4.1 Introduction

In the previous chapter, we experimented with overlap measures exploring character, lexical and lexico-semantic features in order to identify paraphrase pairs, mainly on the Microsoft Research Paraphrase Corpus (MSRPC).

In this chapter, our focus is on a different type of dataset than the MSRPC; a collection of short texts from Twitter. This collection contains highly colloquial language and many NLP tools that assume formal writing cannot parse this kind of non-canonical, written text. Therefore, our knowledge-lean approach may be particularly applicable to this type of data.

We present an approach to identifying Twitter paraphrases using simple lexical overlap features in a further scenario that explores the applicability of knowledge-lean techniques to paraphrase identification. Before that, we present our preliminary results with similarity measures from Chapter 3 and vector operations based on word occurrence in a high dimensional space.

Later, we utilise features based on overlap of words and character n-grams, and train support vector machines (SVMs). Our results demonstrated that character and word-level overlap features in combination can give performance comparable to methods employing more sophisticated NLP processing tools and external resources. We achieved

the highest F-score for identifying paraphrases from the Twitter Paraphrase Corpus as part of the SemEval-2015 Task1[11].

This chapter shows the gradual improvement from similarity measures to SVMs on the Twitter Paraphrase Corpus. Our best performing methods are then also used to experiment with the other two datasets: MSRPC and PAN. The results are analysed in comparison to TPC results.

4.2 Twitter Paraphrase Corpus and SemEval-2015 Task 1

The Semeval-2015 Task1, "Paraphrase and Semantic Similarity in Twitter" involves predicting whether two tweets have the same meaning. Training and test data are provided in the form of a Twitter Paraphrase Corpus (TPC) (Xu, 2014). The TPC is constructed semi-randomly and annotated via Amazon Mechanical Turk by five annotators. It consists of around 35% paraphrases and 65% non-paraphrases. Training and development data consists of 18K tweet pairs and test data consists of 1K pairs. Test data is drawn from a different time period and annotated by an expert. In the next chapter, we will look into the construction process of the TPC and MSRPC in more detail in order to compare them to the paraphrase corpora that we recently constructed for the Turkish language.

A novel aspect of the TPC compared to other paraphrase corpora is the inclusion of topic information, which is also used during the construction process.

4.3 Knowledge-Lean Approaches for Twitter

Colloquial usage of language makes its analysis – whether semantic or syntactic analysis – impractical for commonly used NLP tools. For instance, widely used part-of-speech analysers will not work on Twitter data because sentences such as, *"hiller with the ol glove save"* or *"wtf chris kelly from kris kross dead"* might not be analysed correctly, unless they are normalised to some extent. Construction of a new tool cannot reach the speed of language variety, which is being changed swiftly every day by adding new words, altering existing ones and so on. An effective approach will be one independent from experimental data and adjustable for similar problems.

[11] http://alt.qcri.org/semeval2015/task1/

4.3.1 Pre-Processing

Text pre-processing is essential to many NLP applications. It may involve tokenising, removal of punctuation, part-of-speech tagging, and so on. For identifying paraphrases, this may not always be appropriate. Removing punctuation and stop words, as commonly done for many NLP applications, arguably results in the loss of information that may be critical in terms of paraphrase identification. We therefore keep text pre-processing to a minimum: unless stated otherwise, we lowercased all words from the TPC in experiments completed in this chapter.

The TPC is already tokenised (O'Connor, Krieger, & Ahn, 2010), part-of-speech tagged (Derczynski, Ritter, Clark, & Bontcheva, 2013), and named entity tagged (Ritter, Clark, Etzioni, & Etzioni, 2011). Here, we only experiment on tokenised data, ignoring part-of-speech and named entity tag information. Also note that the corpus was given free from punctuation. In total, training, development and test sets include 13,063, 4,727 and 972 sentence pairs, respectively. As suggested by the task organisers, debatable and discarded pairs are removed from data. Discarded sentences are those sentences of a pair that are exactly identical. Table 4-1 shows the number of pairs in each set of Twitter data after removing debatable and discarded sentence pairs. These have been used in all experiments throughout this thesis.

Data	**Number of pairs**
Train set	11,513
Development set	4,139
Test set	838

Table 4-1: Total sentence pairs in each set of TPC after removing debatable and discarded pairs

A particular issue in dealing with Twitter is the use of capitalisation. Variability in the use of capitals (some tweets may not apply capitalisation rules, others may be written all in uppercase) presents a problem for simple lexical overlap measures between candidate paraphrase pairs. To help overcome this, tokenised tweets are lowercased. Although this potentially causes confusion between proper nouns and common nouns (e.g. *apple* the fruit v. *Apple* the company) our experimental work shows that it most likely increases the quantity of identified paraphrase pairs.

Tweets tend to have a higher proportion of non-literarily written texts and it will be very difficult to construct a Twitter lexicon with good coverage. Due to the character limit, words are often shortened or abbreviated and standard-spelling rules are ignored. In addition, characters may be added for emphasis. Nevertheless, we have not normalised the original texts to compensate for this.

4.3.2 Baselines

All our preliminary experimentations were completed on the development set. The baselines for the development set and three different baselines for the test set (Table 4-2) are computed and provided by Xu et al. (2015) based on the number of pairs shown in Table 4-1. Logistic regression, which is a re-implementation from Das and Smith's (2009) work, was computed on both development and test set. Weighted Textual Matrix Factorization (WTMF) is an improvised version of Guo and Diab's (2012) sentence modelling method.

Baselines	Accuracy	Precision	Recall	F-score
Development Set				
Logistic Regression.	72.55	70.40	38.92	50.13
Test Set				
Random	---	43.4	19.2	26.6
WTMF	---	45.0	66.3	53.6
Logistic Regression	---	67.9	52.0	58.9

Table 4-2: Baseline results from the development and the test set of TPC

4.3.3 Preliminary Experiments

In order to identify a best set of features, we have tried several experiments that need to be mentioned briefly in order to show how we proceed through our main approach. Our first experiment adjusts the similarity measures used in Chapter 3 (Section 3.4.1). Next, we performed logical operations between binary vector operations, where the sentences are represented as binary vectors according to the word occurrence in the experimental data.

4.3.3.1 Experiments with Similarity Measures

Similarity measures adopted from Chapter 3 are applied to the TPC by tuning a threshold on training set. We chose the best threshold for the training set and apply this threshold for

the development set. Hence, Table 4-3 shows the results from similarity measures based on the development set.

The highest results are obtained with lexical features on Jaccard Generalization measure giving an F1-score of 55.1, which is notably higher than the baseline. Although the highest character bigram results are also obtained with Jaccard Generalization, the F-score is very close to the baseline, but not higher. In general, it is safe to say that overlap measures perform better using lexical features than character bigrams.

	Dice		**Cosine**		**Jaccard_Gen**		**Jaccard**	
Features	**Acc.**	**F_Sc.**	**Acc.**	**F_Sc.**	**Acc.**	**F_Sc.**	**Acc.**	**F_Sc.**
Lexical	74.0	54.7	73.3	48.5	74.1	55.1	74.0	54.8
Character Bigrams	73.5	49.2	72.8	46.2	74.1	50.7	73.4	48.9

Table 4-3: Development set results of similarity measures using lexical and character bigram features on the TPC

4.3.4 SVM Classifiers

4.3.4.1 SVM kernels

A Support Vector Machine (SVM) classifier maps the feature vectors into high dimensional vector space separating them into categories and then, given new examples are predicted according to which category they fall under. Its applicability to both linear and non-linear systems has been proven for different NLP applications as well as for paraphrase identification methods. We used Support Vector Machines (SVMs) implementations from scikit-learn (Pedregosa et al., 2001) and experimented with a variety of classifiers. We report here the results obtained using Support Vector Classifier (SVC), which was adapted from *libsvm* (Chang & Lin, 2011) by embedding different kernels. We experimented with linear and Radial Basis Function (RBF) kernels. Linear kernels are known to work well with large datasets and RBF kernels are the first choice if a small number of features is applied (Hsu, Chang, & Lin, 2008); both cases apply to our experimental datasets. Both linear and RBF classifiers are used with their default parameters. Default parameters were used because the main focus here is on the choice of features and representation of paraphrase pairs for classification. Tuning of the SVM parameters to further enhance performance is a secondary issue.

4.3.4.2 Feature Scaling

Scaling is a smoothing technique for data distribution, which transforms features into a new form without loss of their informative characteristics. It is stated (Hsu et al., 2008) that feature scaling is an essential step for SVM classifiers. The SVM features that represent different properties of the data should be scaled (normalised or standardised) for better performance. In order to weigh each feature equally, the numeric range difference of each feature should be reduced to [0,1] or [-1,1]; if not, features that are greater in numeric range will dominate the smaller ones. Weighting features with a metric such as TF-IDF gives a value to each feature according to their significance to text. Ji and Eisentein's (2013) new TF-KLD metric is an improvised form of the TF-IDF metric, used in conjunction with a linear SVM classifier, with which they therefore achieve good performance on the MSRPC. This also proves that although SVM classifiers are powerful for separating features linearly, a successful feature scaling and weighting process increases the results on a given dataset.

However, we kept the scaling as simple as possible by applying a simple scaling mechanism. It is a form of standardisation also called "z-score" in statistics, in which the transformed data variable has a mean of 0 and a variance of 1. Subtracting the mean, μ_x, from the feature vector, x, and dividing each of those features by its standard deviation, σ, scales features, and a new feature vector, $\hat{x}$, is obtained (Equation 4.1). Apart from this *Simple Scaler*, features are kept as they are.

$$\hat{x} = \frac{x - \mu_x}{\sigma} \qquad (4.1)$$

Another alternative scaling technique is to transform each feature's value within a range, such as (0,1). This traditional technique is known Minimum-Maximum Scaling. The minimum feature value, x_{min}, is subtracted from each feature, x_i, and divided by the value difference between the maximum feature value, x_{max} and x_{min} (Equation 4.2).

$$x_{scaled\,[0,1]} = \frac{x_i - x_{min}}{x_{max} - x_{min}} \qquad (4.2)$$

Although Simple Scaler is the only normalisation technique that we used with SVM in this thesis, it is worth mentioning that Minimum-Maximum Scaler (MM_Scaler) has been tried for a few experiments, for reference only, and it has performed well (**Error!**

eference source not found.). There are plenty of scaling techniques that can be tried, but we persist in keeping them as simple as can be so that this simple step will not muddle the main task by introducing a variety of options. Keeping in mind the performance of MM Scaler, we choose to use Simple Scaler for our experiments.

	Linear Kernel				**RBF Kernel**			
Character bigrams	Acc.	Pre.	Rec.	F-sc.	Acc.	Pre.	Rec.	F-sc.
No Scale	87.0	0.8	50.3	61.8	85.2	72.6	46.9	56.9
Simple Scaler	85.3	64.3	66.9	65.6	86.2	67.1	66.3	66.7
MM Scaler	87.4	72.0	64.6	68.1	87.2	71.8	64.0	67.7

Table 4-4: Results from character bigrams using SVM (Linear and RBF kernels) with/without scaling

Table 4-4 shows the results from character bigrams (which will be explained later) in order to present the difference in the results obtained with and without scaling. The dramatic decrease in F1-score results with no scaling shows that scaling is the requirement for increasing the performance of SVM classifiers. Moreover, a different scaling mechanism can affect the results significantly – as can be seen when comparing the MM Scaler results, which are higher than those of the other two in terms of their F1-score. It is safe to say that the performance of RBF kernel increases dramatically when Simple Scaler is used. The performance of the linear classifier, although it performs quite well with scaling, is not affected as much as the performance of the RBF Kernel.

4.3.5 Vector Operations and Boolean algebra

Vector representations are popular because they have been shown to be an effective way of representing documents for many kinds of NLP task. As we know, paraphrase data is constituted only of aligned sentence pairs; it is not reasonable to construct a vector space model of a traditional term-document relationship from each pair. It is more plausible to instead construct binary vectors of sentences based on word occurrence. The binary representation of word occurrence does not take into account how many times a word occurs in a sentence, it only concerns whether or not that word occurs in a sentence.

Logical operations are a conventional way of structuring valid arguments in an abstract level using the truth-values True and False. Arithmetic operations can apply to the

logical operations, but the outcome is one of the two variables of truth-values. We hypothesised that representation of a pair of sentences as binary vectors can be computed with logical operators.

We adopted the idea of accepting that two word vectors are similar if they are close enough to each other in a vector space model. We transform the idea from words to sentences by representing every sentence of a pair as a binary vector. The size of each vector is defined within the size of the corpus experimented on. We perform an element-wise computation between two vectors of sentences using the three logical operators AND, OR and XOR. The truth tables of these operators are shown in Table 4-5. Owing to the fact that paraphrasing requires bidirectional relations, we only experimented with the operators whose outcome remains the same in accordance with the order of the vectors. For instance, the logical operator NOT is excluded because two different outputs of a pair are obtainable, due to its asymmetric notion.

AND			OR			XOR		
1	1	True	1	1	True	1	1	False
1	0	False	1	0	True	1	0	True
0	1	False	0	1	True	0	1	True
0	0	False	0	0	False	0	0	False

Table 4-5: Truth tables of Logical Operations: AND, OR and XOR

A model is constructed for these operations based on word occurrence, calculated considering a whole dataset of unique words. The next step is to carry out operations between the two binary vectors of source sentence (S_1) and target sentence (S_2) of a pair. The vectors are of dimension *d,* where *d* is the size of the vocabulary of the corpus. We obtained two different vectors, V_{s1} and V_{s2}, one for the source sentence and the other for the target sentence. These two vectors are obtained based on the presence (1) or absence (0) of a word (*w*). Sentence vectors are denoted as V_{s1} or V_{s2}, represented as binary vectors.

$$\overrightarrow{V_{s1}} = (w_1 = 1, w_2 = 1, w_{3=}0 \dots, w_{d=}0); \quad \overrightarrow{V_{s1}} = (w_1 = 0, w_2 = 1, w_{3=}0 \dots, w_{d=}1)$$

Logical operations between these two binary vectors are then calculated for each pair, where each pair constitutes two vectors that become a new vector represented with the dimension of *d,* which is the size of the vocabulary. This means the size of a new vector

will not change, and every element can be used as a feature of SVMs. Each dimension of these new vectors is used as an input feature for SVM classifiers.

One aspect of using SVM is to scale the features in order to keep them within a range of numbers, which loses its importance in logical operations because the features are constituted of merely 1 and 0 values.

4.3.5.1 Development Set Results

Results from each logical operation with a linear classifier are shown in Table 4-6.

	SVC (Linear Kernel)			
Features	**Acc.**	**Pre.**	**Rec.**	**F_Sc.**
AND	64.5	49.7	09.5	16.0
OR	63.2	46.4	24.7	32.2
XOR	61.5	45.5	43.1	44.2

Table 4-6: Results from logical operations AND, OR and XOR

As can be seen, the results are below the baseline. It clearly shows that these operations are not suitable for separating the paraphrase and non-paraphrase pairs. The fact that AND operators take into account the words present in the sentences of a pair, while OR operators ignore the dissimilar words, does not really help to classify the sentence pairs, whereas the XOR operator gives equal weight to features having two False and two True, unlike OR and AND operators. However, the XOR operator performs well compared to the others; it performs more like a dissimilarity function, representing the joint absence of word as true.

4.3.6 Deviating from Set Theory: Four Features and Instances

As the basis for deriving a number of overlap features, we consider different representations of a text as a set of tokens, where a token may be either a word or a character n-gram. For the work described here we restrict attention to word and character unigrams and bigrams. Use of a variety of machine translation techniques (Madnani et al., 2012) that utilise word n-grams motivated their use in representing texts for this task. In particular, word bigrams may provide potentially useful syntactic information about a text. Character bigrams, on the other hand, allow us to capture similarity between related word forms. Possible overlapping features are constructed using the following basic set operations.

Size of union: the size of the union of the tokens in the two texts of a candidate paraphrase pair.

Size of intersection: the number of tokens common to the texts of a candidate paraphrase pair.

Text size: the size of the set of tokens representing a given text.

A pair of tweets constitute two sentences. Each pair is represented with a set of four features denoted with *U, N, L1,* and *L2*. ***U*** is the *size of union*, ***N*** is the *size of intersection*. The features ***L1*** and ***L2*** are the length of sentence 1 and length of sentence 2, respectively, of a candidate paraphrase pair. These four features are computed for word and character n-grams (up to two) features. This yields a total of eight possible overlap features for a pair of texts, plus four ways of measuring text size. Each data instance is a vector of features representing a pair of tweets. In order to select an optimal set of features we ran a number of preliminary experiments.

Intuitively, knowing about the union, intersection or size of a text in isolation may not be very informative. However, for a given token type, these four features in combination provide potentially useful information about similarity of texts. In the following, C1 and C2 each denote four features (U, N, L1 and L2) produced by character unigrams and bigrams, respectively. Similarly, W1 and W2 denote the four features generated by word unigrams and bigrams, respectively. Combinations (e.g. C1W2) represent eight features: those for C1 plus those for W2.

Table 4-7 demonstrates the results from the development set that led us to decide which n-gram model performs best individually and in combination. We present results obtained both from the linear kernel and the RBF kernel. The best performance is achieved with C2 individually and in combinations with C2W1. Although W2 solely performs below the baseline, the second highest results are obtained in C2W2. A slight increase was observed in C1W1, notwithstanding that C1 features are not informative at all. This shows that using these features are interdependently works well.

SVM (linear kernel)					SVM (RBF kernel)				
Features	**Acc**	**Pre.**	**Rec.**	**F-sc.**	**Features**	**Acc**	**Pre.**	**Rec.**	**F-sc.**
C1	64.5	0.0	0.0	0.0	C1	66.4	57.1	21.8	31.6
C2	74.5	70.2	48.4	57.7	C2	74.6	72.	47.	56.5
C1C2	74.5	70.3	48.5	57.4	C1C2	74.1	71.5	44.8	55.1
W1	74.1	70.5	46.5	56.0	W1	74.3	73.7	42.8	54.2
W2	70.5	63.9	38.8	48.3	W2	70.7	64.6	38.6	48.4
W1W2	74.0	69.9	46.9	56.2	W1W2	74.1	74.	41.4	53.1
C1W1	74.2	70.4	47.2	56.5	C1W1	74.1	71.9	44.2	54.7
C2W2	74.9	71.1	49.1	58.1	C2W2	74.8	72.6	46.8	56.9
C1W2	71.4	72.0	31.9	44.2	C1W2	71.3	68.3	35.7	46.9
C2W1	75.6	72.4	50.6	59.6	C2W1	75.7	74.	48.8	58.8
Baseline	72.6	70.4	38.9	50.1	Baseline	72.6	70.4	38.9	50.1

Table 4-7: U, N, L1 and L2 features operating on individual and combined character and word units; Results obtained using SVM (Linear and RBF kernels) on the development set of TPC

4.3.6.1 Extending n-gram models

From the previous results we have seen that the scores of word n-grams decrease from unigrams to bigrams, in contrast to the behaviour of character n-grams. This means there is room to explore character n-gram models – from trigrams onwards until there is a downturn in the results (Table 4-8).

SVM (linear kernel)					SVM (RBF kernel)				
Features	**Acc**	**Pre.**	**Rec.**	**F-sc.**	**Features**	**Acc**	**Pre.**	**Rec.**	**F-sc.**
C3	75.33	72.09	49.73	58.86	C3	75.18	73.07	47.62	57.66
C4	75.11	73.63	46.53	57.02	C4	75.13	73.36	46.46	57.01
C2C3	75.23	71.76	49.8	58.8	C2C3	75.25	72.82	48.3	58.08
C3C4	74.8	70.7	49.6	58.3	C3C4	75.4	73.5	47.8	58.0

Table 4-8: U, N, L1 and L2 features operating on individual and combined character bigrams, trigrams and four-grams; Results obtained using SVM (Linear and RBF kernels) on the development set of TPC

The set of four features of character trigrams, which is denoted as C3, individually, showed a notable increase compared to character bigrams. We stopped experimentation after character four-gram (C4) models, where we observed a downturn in results. In addition to the individual results from C3 and C4, the combination of C2 and C3 are experimented with in order to establish whether the scores get higher when they are combined. The combination C2C3 does not increase performance, and the results are even lower with C3C4 as compared to the individual results obtained from C3.

4.4 SVM Classification and Results

In the previous section, we showed the development results, which are used as an indicator for validating the test set. Here, we will provide a fine grain analysis of the performance of the set of four features in combination and individually. Then, the official results from the SemEval-2015 Task1 will be shown in comparison to the best-performing methods on the paraphrase identification task.

4.4.1.1 Feature Ablation

We performed feature ablation to see which features among the set of four features (U, N, L1 and L2) work well, individually and in combination with others. Our basic set of four features has been experimented with individually; each individual feature was computed based on a pair of sentences. *U* is the number over both sets of unique tokens; *N* is the number of common tokens that occur in both sentences in a pair. The other two features, lengths of sentence 1 and sentence 2 of a pair (represented as *L1* and *L2*) are separate features, but because these features do not represent attributes of a pair when looked at individually, we always use these two features together during feature ablation. In Table 4-9, although each feature is experimented with individually, we only show relevant results that prove the individually combined features of C2 and W1 work better than the set of four features. The highest results obtained with linear and RBF kernels are highlighted in grey. Comparing the results of individual features in C2 and W1, C2 features mostly perform better than W1.

Ablation	Classifiers➔	SVM (linear kernel)				SVM (RBF kernel)			
	N-gram features	Acc	Pre.	Rec.	F-sc.	Acc	Pre.	Rec.	F-sc.
Individual features of C2 and W1	$C2_{U,N}$	85.2	63.4	69.1	66.1	85.3	63.7	69.1	66.3
	$C2_{U,L1,L2}$	86.3	67.4	66.3	66.9	87.0	70.9	64.0	67.3
	$C2_{N,L1,L2}$	84.8	62.8	67.4	65.0	86.0	66.7	66.3	66.5
	$W1_{U,N}$	84.6	62.2	66.9	64.5	85.4	65.5	64.	64.7
	$W1_{N,L1,L2}$	85.4	64.6	66.9	65.7	86.5	71.5	58.9	64.6
	$W1_{N}$	85.8	67.5	61.7	64.5	85.8	67.5	61.7	64.5
Combined C2 and W1	$C2_{all}W1_{N,L1,L2}$	85.8	64.4	71.4	67.8	86.2	66.1	69.1	67.6
	$C2_{all}W1_{U,N}$	86.5	68.0	66.9	67.4	85.7	64.9	68.6	66.7
	$C2_{U,N}W1_{N}$	85.6	64.1	70.3	67.0	85.6	64.1	70.3	67.0
	$C2_{U,N}W1_{all}$	85.8	64.9	69.7	67.2	85.3	63.8	68.6	66.1
	$C2_{U,L1,L2}W1_{N}$	85.9	65.1	70.3	67.6	86.6	67.8	68.6	68.2
	$C2_{U,L1,L2}W1_{N,L1,L2}$	86.3	66.0	70.9	68.2	86.5	68.0	66.9	67.4

Table 4-9: Feature ablation results with character bigrams and word unigrams on the test set of TPC

In order to visualise how these individual features are separable with a hyperplane, we plotted a few comparable features together from the test set using Matplotlib[12] software (Hunter, 2007). Figure 4-1 shows the union and intersection features of character bigrams with two different classifiers. The two classifiers are able to find a hyperplane that separates the paraphrases based on the two features union and intersection. These values give high confidence when used together and their confidence level is evidently not related to the type of classifier used. Conversely, lengths of the sentence values are not enough to separate the instances, because these features (L1 and L2) do not represent the sentence pair when experimented with individually; they only can be used to support the two other features (Figure 4-2).

[12] http://matplotlib.org/

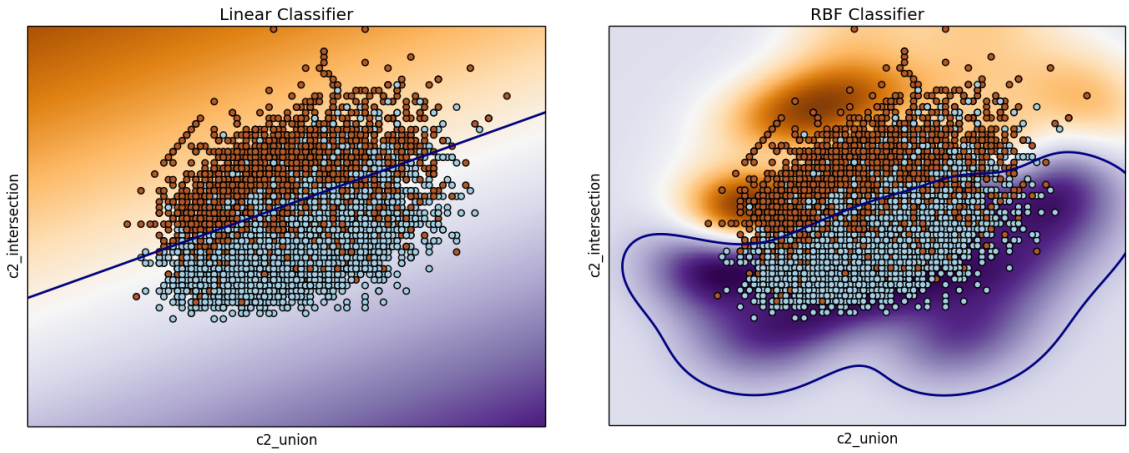

Figure 4-1: Character bigrams of union and intersection features of Linear and RBF classifiers on the test set of TPC

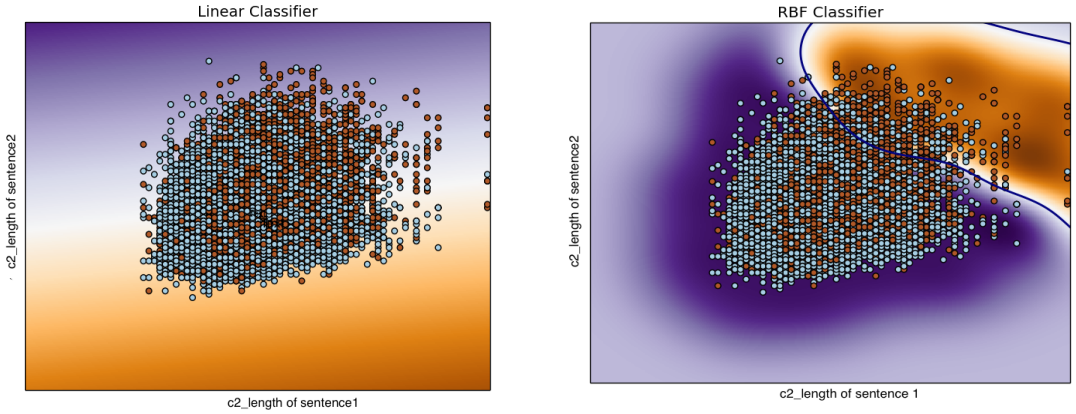

Figure 4-2: Inseparable character bigrams of length features (L1 and L2) with Linear and RBF classifiers on the test set of TPC

4.5 Test Set Results

By the time we entered the SemEval-2015 Task 1 (Xu et al., 2015), our experimentation was not complete, as indicated in our article (Eyecioglu & Keller, 2015). We only had the results of n-gram models up to 2 and the exploration on feature ablation was not finalised. Table 4-10, below, shows the results that are officially demonstrated on SemEval-2015 Task1 (See Appendix B for all participating team results and our team name is "ASOBEK"). Here, we only show the highest 3 results. The highest results obtained after feature ablation (Table 4-9) are also included.

Model	**Acc.**	**Pre.**	**Rec.**	**F-sc.**
$C2_{U,L1,L2}W1_N$ (RBF kernel)	86.6	67.8	68.6	68.2
SVM (RBF kernel)	85.7	64.9	68.6	66.7
Model	**Acc.**	**Pre.**	**Rec.**	**F-sc.**
01_svckernel (ASOBEK)	86.5	68.0	66.9	67.4
(Zarrella, Henderson, Merkhofer, & Strickhart, 2015)	--	56.9	80.6	66.7
(J. Zhao & Lan, 2015)	--	58.3	76.7	66.2
Baseline	--	67.9	52.0	58.9

Table 4-10: The official results of SemEval-2015-Task1 (the highest 3 results) on PI task

Our linear classifier uses only six features having applied feature ablation on word unigram features: set of four features of character bigrams (U, N, L1, L2) combined with union and intersection features (U, N) of word unigrams, which would have ranked it 1st in the SemEval-2015-Task1. Zarrella et al. (2015)'s method ranked 2nd and their method is a combination of seven different approaches, called "seven systems". This seems to demonstrate that knowledge-lean methods can perform as well as, or even better than, more sophisticated approaches. Zhao and Lan's (2015) method uses deep learning as well as a variety of linguistic features. Besides using a normalisation dictionary, WordNet was also used for looking up synonymous relation between words.

A more detailed set of results extended up to four character n-grams is shown below. Table 4-11 shows the test set results from four features of each n-gram model (individually and in combination), in order to see whether there is a correlation between development and test results. The best results are obtained from the combination of all C2 and W1 features with the linear kernel. Although the accuracy of C12W12 is higher, its F-score is slightly lower than that of C2W1.

Between the results of the individual features (C1, C2, W1 and W2), the lowest results are obtained with C1; for the linear kernel C1 is not able to separate paraphrases at all. The performance of C2 features is better than the other individual features with both linear and RBF kernels.

SVM (linear kernel)					SVM (RBF kernel)				
Features	**Acc.**	**Pre.**	**Rec.**	**F-sc.**	**Features**	**Acc.**	**Pre.**	**Rec.**	**F-sc.**
C1	NA	NA	NA	NA	C1	76.4	40.7	28.6	33.6
C2	85.3	64.3	66.9	65.6	C2	86.2	67.1	66.3	66.7
W1	85.0	63.2	66.9	65.0	W1	85.2	66.5	58.9	62.4
W2	85.0	65.8	58.3	61.8	W2	85.0	65.8	58.3	61.8
C1C2	85.2	63.5	68.6	65.9	C1C2	86.0	66.7	66.3	66.5
W1W2	83.9	60.3	66.9	63.4	W1W2	85.9	69.1	58.9	63.6
C1W1	84.0	60.9	65.7	63.2	C1W1	85.2	64.9	63.4	64.2
C2W2	85.8	65.4	68.0	66.7	C2W2	85.3	65.3	63.4	64.4
C1W2	84.8	69.1	49.7	57.8	C1W2	84.6	66.4	53.1	59.1
C2W1	85.9	65.2	69.7	67.4	C2W1	85.4	65.1	65.1	65.1
C12W12	85.7	64.7	69.1	66.9	C12W12	86.0	66.9	65.7	67.0

Table 4-11: U, N, L1 and L2 features operating on individual and combined character and word units; Results obtained using SVM (Linear and RBF kernels) on the test set of TPC

Following this, trigram and four-gram results are shown in Table 4-12. There is a notable increase in overall results compared to the previous table (Table 4-11). The lowest results are obtained with C4 – although they are still higher than C1 and W2 – but its combination with C3 using the linear kernel reaches the highest F-score. However, overall Table 4-12 shows that the results are still slightly lower than the C2W1 results from Table 4-11 (with the exception of C3C4 features with the linear kernel).

SVM (linear kernel)					SVM (RBF kernel)				
Features	**Acc**	**Pre.**	**Rec.**	**F-sc.**	**Features**	**Acc**	**Pre.**	**Rec.**	**F-sc.**
C3	85.8	64.9	69.7	67.2	C3	85.9	66.1	66.9	66.5
C4	85.3	65.1	64.0	64.6	C4	84.8	63.8	63.4	63.6
C2C3	85.9	65.4	69.1	67.2	C2C3	85.8	65.4	68.0	66.7
C3C4	86.2	65.6	70.9	68.1	C3C4	86.2	66.5	68.0	67.2
C1234	85.4	63.9	69.7	66.7	C1234	86.2	66.5	68.0	67.2

Table 4-12: U, N, L1 and L2 features operating on individual and combined character n-grams (up to 4); Results obtained using SVM (Linear and RBF kernels) on the test set of TPC

4.6 Comparison to MSRPC and PAN

In order to see how robust these results are, the same methodology is applied to other paraphrase corpora, to find out whether these features are generally applicable. This section explains how the same methodology was adjusted for different paraphrase corpora, the MSRPC and PAN, from that used for the TPC results.

4.6.1 Pre-processing

We used the RASP Toolkit (Briscoe et al., 2006) to perform first tokenisation then lowercasing on both the MSRPC and PAN. In contrast to the TPC dataset, the sentences of

the MSRPC and PAN conform more to the standard rules of the English language. They include punctuation, grammar rules, etc., which were kept as they are – we experimented on merely lowercased datasets.

4.6.2 SVM with 10-fold Cross-Validation on the MSRPC and PAN

The fact that the TPC has a development set for validation of the results, while the MSRPC and PAN only have test sets, might compromise our comparison if an extra validation step is not applied. Bearing in mind that our results might be slightly lower than expected with 10-fold cross validation, it is still one of the most plausible ways to compare our results reliably.

A 10-fold cross validation was applied before the classification process. We gathered the whole dataset (train/test) and calculated the feature values. These features were split into 10 different sets after applying simple scaling. Each of 10 sets is used as test set against the remaining 9 sets, on which the classifiers are trained. Hence, the mean of the 10 results that are obtained from each of 10 sets is accepted as the final result. Both the linear and RBF kernels of SVM are experimented with for character and word n-gram models (up to 2), in the same way as the TPC results were obtained. Although we obtained the highest results through a process of feature ablation, we have not experimented further with any feature ablation on the MSRPC or PAN.

4.6.3 Results

Table 4-13 and Table 4-14 demonstrate the results that are obtained from 10-fold cross validation of SVM classifiers on the MSRPC and PAN, respectively. We used the same notations of n-grams that we used for the TPC results. For instance, C2 stands for character bigrams including the set of four features, while C2W1 stands for the combinations of character bigrams and word unigrams including their set of four features, making eight features in total.

SVM (Linear kernel)					SVM (RBF kernel)				
Features	**Acc**	**Pre.**	**Rec.**	**F-sc.**	**Features**	**Acc**	**Pre.**	**Rec.**	**F-sc.**
C1	69.8	69.8	97. 0	81.2	C1	69.8	69.8	97. 0	81.2
C2	72.8	75.4	88.5	81.4	C2	73.0	73.6	93.4	82.3
C1C2	72.7	74.4	90.7	81.7	C1C2	72.9	73.5	93.2	82.2
W1	72.4	74.0	90.9	81.6	W1	73.1	74.8	90.5	81.9
W2	67.2	67.2	1.0	80.4	W2	69.6	70.1	95.6	80.9
W1W2	72.8	74.6	90.3	81.7	W1W2	73.1	74.9	90.3	81.9
C1W1	72.6	73.5	92.7	82.0	C1W1	73.1	74.0	92.5	82.2
C2W2	73.3	75.3	89.7	81.9	C2W2	74.0	75.0	92.2	82.7
C1W2	70.4	70.7	95.4	81.2	C1W2	71.2	71.7	94.5	81.5
C2W1	73.5	75.9	89.0	81.9	C2W1	74.2	75.3	91.6	82.7
C12W12	74.0	75.7	90.2	82.3	C12W12	74.2	75.3	91.7	82.7

Table 4-13: U, N, L1 and L2 features operating on individual and combined character and word units; Results obtained using 10-fold cross validation with SVM (Linear and RBF kernels) on MSRPC

Table 4-13 shows that the highest F-score was obtained using character bigrams and word unigrams (C2W1) with the RBF kernel. Combining all 16 features (C12W12) does not change the results at all, which means the combinations of these features do not add any performance over C2W1. However, the linear classifier reaches the highest point when all 16 features are combined (C12W12). Also, word bigrams (W2) give the lowest results with both classifiers, which shows that these features do not perform well, either when used alone or when used in combination with other features.

SVM (Linear kernel)					SVM (RBF kernel)				
Features	**Acc**	**Pre.**	**Rec.**	**F-sc.**	**Features**	**Acc.**	**Pre.**	**Rec.**	**F-sc.**
C1	72.7	73.1	72.0	72.5	C1	76.1	78.5	72.0	75.1
C2	89.7	91.2	87.9	89.5	C2	90.9	92.1	89.6	90.8
C1C2	89.9	91.2	88.4	89.8	C1C2	90.8	91.5	90.0	90.7
W1	89.8	90.1	89.5	89.8	W1	92.0	93.0	90.8	91.9
W2	89.1	89.2	89.0	89.1	W2	90.1	91.1	88.8	89.9
W1W2	90.2	90.5	89.8	90.2	W1W2	92.1	93.0	91.1	92.0
C1W1	90.3	90.7	89.8	90.3	C1W1	91.8	92.7	90.7	91.7
C2W2	90.4	91.6	88.9	90.3	C2W2	91.9	92.8	90.8	91.8
C1W2	89.3	89.9	88.7	89.3	C1W2	90.2	91.4	88.9	90.1
C2W1	90.4	91.5	89.0	90.2	C2W1	92.4	93.4	91.3	92.3
C12W12	90.8	91.8	89.6	90.7	C12W12	92.3	93.1	91.5	92.3

Table 4-14: U, N, L1 and L2 features operating on individual and combined character and word units; Results obtained using 10-fold cross validation with SVM (Linear and RBF kernels) on PAN

The results from Table 4-14 are comparable to the TPC results in terms of the lowest results that were obtained from character unigrams (C1). A significant finding is that the RBF kernel outperforms the linear kernel remarkably, although this is not the case with

MSRPC. The highest F-score is obtained from character bigrams and word unigrams (C2W1) with the RBF kernel, and combining all 16 features does not change the results, like on MSRPC.

We compared two different paraphrase corpora using 10-fold cross validation experiments with SVM of two different classifiers. Although, the MSRPC and PAN are quite similar in contrast to TPC, these features outperformed against TPC, and performed relatively well on the other datasets.

4.7 Analysis and Discussion

Examining overlap of character bigrams was more informative than for character unigrams. We hypothesise that measuring overlap of character bigrams provides a way of detecting similarity of related word-forms. It thus performs a similar function to stemming or lemmatisation in other language processing tasks, whilst retaining some information about difference. This may be especially helpful with Twitter, where a variety of idiosyncratic spellings and shortened forms may be observed alongside the usual morphological variants. In addition, character bigrams' combination with word unigram features increased the performance on all datasets.

A strength of our approach is that pre-processing is kept to a minimum. This may explain why our system outperforms the systems entered for the SemEval-2015 Task 1, and most paraphrasing identification methods utilise a similar set of overlap features. Methods that require the removal of stop words, OOV words etc. seem to lose potentially useful information. On the other hand, we found that normalising tweets with regard to capitalisation enhanced the performance of the classifier.

A few example pairs are presented below that cannot be identified correctly by the system we used in SemEval-2015 Task 1:

Paraphrase pairs that are predicted false
• *the ending to 8 mile is my fav part of the whole movie* • *those last 3 battles in 8 mile are the shit*
• *chris davis is putting the team on his back* • *chris davis is so fucking good*
Non-Paraphrase pairs that are predicted true
• *hahaha that sounds like me* • *sounds like a successful day to me*
• *world of jenks is on at 11* • *world of jenks is my favorite show on tv*
Linear Kernel can identify but not RBF Kernel
• *damn jason kidd and grant hill retiring together* • *jason kidd got his ring with the dallas mavericks*
• *new ciroc amaretto i need that* • *new ciroc flavor on the market gotta try that shit*

Throughout the experimentation of our methods, there have been a few suggestions, which might help to increase the performance of the applied methods. For instance, we have suggested that the feature scaling is an essential part of SVM classification. In addition, feature weighting might be considered.

We have discovered the four basic set features (N, U, L1, L2) and our experimental results are competitive on three different paraphrase corpora. As shown in Chapter 3, these features did not perform well with similarity measures, but outperformed many other more sophisticated methods when used with SVM classifiers. This indicates that a knowledge-lean method may be competitive as long as it suits the application and the data used for the experiment.

4.8 Overall Summary

In this chapter, we experimented with a variety of knowledge-lean approaches. We employed similarity measures using simple overlap features, and then, SVMs were applied using two different kernels: Linear kernels and RBF kernels. Preliminary experiments, which do not perform well for paraphrase identification, helped us to reconsider our alternatives; this consequently led us to identify finer features of the set theory.

We presented a knowledge-lean approach to identifying paraphrase pairs using character and word n-gram features by employing SVM classifiers. As well as discovering the applicability of a set of four features (U, N, L1, L2) for paraphrase identification tasks, our applied features are shown to be very informative and work well with SVM classifiers. Moreover, our latest results show that their varied combinations increase the performance of the SVM classifiers. SVM classifiers are one of the most commonly used classification techniques in NLP and in paraphrasing tasks, and they are known to work well with a high number of features. We demonstrated that better results can be obtained using fewer but more informative features. Our solution already outperforms the most sophisticated methods applied on the TPC, and competitive results are obtained on the MSRPC and PAN.

Moreover, performing feature ablation showed that the individually combined features increase the performance of SVM classifiers. However, increasing the number of n-grams to four-grams resulted in lower results than trigrams.

Our results demonstrate that knowledge-lean methods based on character and word level overlap features in combination can give good results in terms of the identification of Twitter paraphrases.

SVM classifiers were successfully used to identify paraphrase pairs given just a few simple features. Using classification based on a tuned threshold does not seem to be as effective as using all features individually as an input to SVM classifiers. Our system performed generally much better (in terms of F-score) compared to other, more sophisticated, participating systems in SemEval-2015 Task 1.

Chapter 5

5 Constructing a Turkish Paraphrase Corpus and Applying Knowledge-Lean Techniques for Paraphrase Identification

5.1 Introduction

Previously, it was seen that usage of simple overlap features of character and word n-grams with the right combination give good results on the Twitter Paraphrase Corpus (TPC) and competitive results on the Microsoft Research Paraphrase Corpus (MSRPC). Moreover, they outperformed many sophisticated methods without the use of any external resource or tool. It follows that these techniques may result in PI for languages that are poor in language processing resources being every bit as accurate as for languages that are resource-rich.

In spite of the large amount of data that can be found and is easily accessible online, there are only a few corpora available in languages other than English for the evaluation of paraphrase identification methods. Therefore, we decided to construct a paraphrase corpus of our own in Turkish in order to contribute to this topic area and language resources for the research community. Turkish is a highly agglutinative language, with richer word-formation processes than English. In terms of paraphrasing, it seems challenging to look at Turkish, because a word can have multiple meanings and can be used in multiple places in a sentence while keeping the same word form. For instance, the word “açık” corresponds to the meaning of the following English words: open, clear, fair, obvious, explicitly, cloudless, visible, light (colour) and so on.

In this chapter, we will give an overview of paraphrase corpora in other languages that we are aware of and step through the construction of the Turkish Paraphrase Corpus in detail. We aim to draw a comparable conclusion between the corpora from two different languages: English and Turkish. Although the languages are very different, we aim to show that paraphrase identification can be done successfully using the same methods. Results are analysed for the constructed dataset and compared to other paraphrase corpora: the MSRPC and TPC. Possible future directions regarding corpora construction and use of overlap features are discussed at the end of this chapter.

5.2 Paraphrase Corpora in Other Languages

Recent research has drawn some attention to the importance of paraphrasing tasks in NLP. This research has been supported through the construction of paraphrase corpora, designed for the investigation of paraphrasing tasks. For English, the MSRPC has become the gold standard data for experimenting with paraphrase identification. Other corpora have followed: PAN and the TPC, which have been discussed in previous chapters. With regard to other languages, paraphrasing research is often still at an early stage. Indeed, this progress is comparatively slow because of limited resources, tools, etc. Despite the fact that the resources of these languages are certainly inadequate, the increasing usage of the internet provides a great deal of user-generated text, and this in turn helps to advance available tools and bring forward their development.

A multi-task approach – for both translation and paraphrasing – was developed by Chen and Dolan (2011). They constructed a video description corpus for paraphrase evaluation: the Microsoft Research Video Description Corpus[13]. This corpus is a collected set of multilingual sentences generated from a collection of short videos that are described in one sentence by Mechanical Turk workers. One recent approach (Ganitkevitch et al., 2013), which is an encouraging attempt regarding paraphrasing, has been completed for multi-lingual paraphrase databases at word and phrase level: PPDB[14]. This rich database was constructed using pivot (paraphrases achieved by translating to a target and then back to source) techniques (Bannard & Callison-Burch, 2005) and was recently expanded into

[13] http://research.microsoft.com/en-us/downloads/38cf15fd-b8df-477e-a4e4-a4680caa75af/
[14] http://paraphrase.org/#/

more than 20 languages. Although it is not at sentence level, at least not yet, a substantial number of paraphrases are available in many languages.

There are some other paraphrase corpora: the TICC Headline Paraphrase Corpus (Wubben, Bosch, & Krahmer, 2010) is a collection of English and Dutch, but it has not been made publicly available yet. The Hebrew Paraphrase Corpus (Stanovsky, 2012) serves the purpose of paraphrase identification and a scoring schema is introduced as paraphrases, non-paraphrases and partial paraphrases. In terms of scoring, partial paraphrases do not fall into the criteria that are defined for the purpose of comparison with other paraphrase identification corpora or sentential semantic similarity. Another collection, of French text, is WiCoPaCo, (Max & Wisniewski, 2010), which includes text normalisation, corrections, relations and so on. However, this data is not constructed solely for paraphrasing tasks, and consequently no scoring scheme is supplied. A Turkish Paraphrase Corpus has been previously constructed (Demir, El-Kahlout, Unal, Kaya, & El-kahlout, 2012). Although this corpus is diversified by gathering four different resources, there are no negative instances, and no scoring schema is available, which makes these data less useful.

5.3 About the Turkish Language

The Turkish language is classified under Turkic languages group of western and central Asia, which belongs to the Altaic language family.

Turkish is a highly inflected and agglutinative language that uses suffixes and affixes commonly. The frequent usage of affixes (a word can have more than one affix) is very typical either to change the meaning or stress the word. Turkish and English are alike in terms of usage of Latin alphabet – except for a few letters. Words are space-separated, but word order follows subject-object-verb, unlike English. In addition, Turkish has the advantage of being separable by white spaces and is case sensitive. This works in our favour regarding the methods that we will experiment with, because we will only use the character and word n-gram features with SVMs, for a fair comparison to the TPC and MSRPC results from Chapter 4.

Our focus is on whether these methods are applicable to another language, and how similarities and differences between the two languages affect the outcomes. The next

section is firstly about construction of Turkish Paraphrase Corpus. In order to avoid confusion with the abbreviation for the Twitter Paraphrase Corpus (TPC), the Turkish Paraphrase Corpus is abbreviated as TuPC. We start experimenting on the TuPC using similarity measures adapted from Chapter 3. These results will be evaluated using the best performing method from Chapter 4. A comparison of the methods and illustration of the results from other paraphrase corpora will then be provided.

5.4 Data Collection Method

As we mentioned in Chapter 2, one of the ways of collecting paraphrase corpora is to crawl the web from daily news sites, where daily events can be clustered with respect to the subject matter being reported (Dolan et al., 2004). Within each cluster, the same event is reported differently, either by reporting details in different wording or expressing them with distinct aspects. Although this type of collected data is abundant, its variability means that it later requires an extra step of refining.

Our data collection strategy combines the methodologies from previously constructed corpora MSRPC and TPC (Brockett & Dolan, 2005; Xu, 2014). We automatically extracted, aligned, and paired sentences from daily news websites. Candidate pairs were hand-annotated by looking up their context. A more detailed description of our strategy is explained in the following section.

5.4.1 Source Data

We crawled the web for a number of daily news sites (Milliyet, Aksam, Posta, Yeniakit, Yenisafak, Meydan Gazetesi, Yenisafak, Radikal, Taraf, Sabah, Star and Sozcu) for a month. This allowed us to collect enough data, due to our deep search. We did not just look at headlines, but also crawled all related sentence-like text, subheadings and paragraphs, since it is more advantageous to crawl a whole text to gain negative samples. The MSRPC and previously collected data focused only on headlines, which results in more positive samples of data than negative samples. The Twitter Paraphrase Corpus has more negative samples because the data is collected through topic features –sentences sharing the same hash tag are collected together. Our approach, in a way, combines these two approaches. We observed that sentences within a paragraph connect to the same topic with more

information or a different aspect of the topic, which makes them suitable for obtaining natural negative paraphrases.

We implemented our own crawler, too. Our crawler is simple but has a viable mechanism for extracting plain text and can cluster according to a pre-selected list of sub-topics. The raw data is then gathered and freed from unwanted non-text content. Figure 5-1 demonstrates the overall steps in crawling the web for plain and clean text gathered from daily newspapers. Each step will be described in detail.

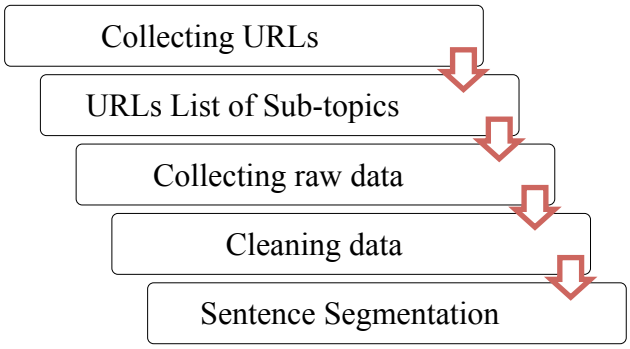

Figure 5-1: Data collection

Firstly, we gathered a list of URLs from a website that links to most daily Turkish newspapers. From each URL, another list of URLs was extracted for each newspaper. We searched for the widely used heading tags by looking at categories on news websites. For instance, the latest news can be found under the heading “last minute” on one of the websites, whereas another website names this “latest”.

[haber, gundem, guncel, sondakika, haberler, turkiye, haberhatti,…,]

A subset of popular headings, shown the list above, was established to extract topic related news. This step serves as an advanced filtering process that limits the topics to within “news, latest, sports” and so on, rather than having news related to “travel”, “fashion” and other miscellaneous texts. These sub links are crawled, to obtain every possible text under the selected categories. For each category, we gathered all news from different news agencies. In Figure 5-2, we show a screenshot of the extracted subheadings of one of the websites, shown on the left.

Figure 5-2: A screenshot of one of the news agency's links

We collected the raw data between 4 and 14 May 2015 and 17 and 23 June 2015. In a very short time, we were able to obtain enough data; we stopped collecting between 14 May till 17 June explicitly because we realised that the websites are not completely updated everyday, at least not every section, and that causes redundancy, which was resulting in duplicate lines in data.

After crawling the news sites, a list of texts within html structures were gathered and cleaned by removing the html format. We implemented a regular expression script: our script removes any related text between every "<" and ">".

The newly collected cleaned dataset consisted of duplicate lines, long paragraphs, titles etc., which needed to be simplified at sentence level. First, we got rid of duplicate lines. Then, a sentence segmentation tool from NLTK (Bird et al., 2009) was trained on a small set of Turkish text, to be used on our dataset. Although this tool was highly successful in capturing sentences by splitting paragraphs into sentences, we used manual correction on samples where the segmentation tool was not able to work properly even after the training process. We believe that the sentence segmenter tool can be trained on a large and representative collection of Turkish text for better results.

Approximately 10,000 lines of texts were clustered on a daily basis. Since we obtained not only headlines but also titles, subtitles and paragraphs, these data included a varied set of sentences – some of them had a title of one word, while in other cases there were long text fragments formed of more than one sentence. Finally, after the segmentation process, we obtained a collection of datasets of aligned texts. We filtered the collected dataset with a few simple mechanisms that will be explained in the next section.

5.5 Filtering the Data and Aligning Sentences for Monolingual Parallel Corpora

The initial collected dataset needed to be aligned as one sentence per line, which was handled by the sentence segmentation tool. Data must then be filtered, and sentences that are to be paired aligned, which requires a well-defined methodology.

The sentence alignment process is a systematic process that is commonly used with Statistical Machine Translation techniques, where alignment is a necessity for the construction of multilingual parallel corpora, where all words or phrases are required to be mapped one-to-one. This step is more important for sentential paraphrases. This is because sentential paraphrases are more complex than any other type of paraphrases. This is sometimes because of rewording that expands or condenses sentences, causing a considerable difference in the lengths of source and target sentences. Rule-based approaches for paraphrasing, such as DIRT rules (Lin & Pantel, 2001), reveal some basic forms of paraphrases, but they are not adequate for covering all paraphrasing rules. Besides, attempting to define a complete set of rules is impractical. Therefore, we have not used any sentence alignment tool; instead we rely on the filtering process that will remove any unwanted sentences, and pair as many sentences as possible as potential paraphrase pairs.

We defined a set of filtering criteria based on previous methods used for paraphrase corpora construction. Our method filters the dataset by removing unwanted items, at the same time aligning sentences as pairs within rules defined in advance:

Sentences are discarded if:

1. They are greater than 40 words long or less than 5.
2. Lexical word overlap between two candidate sentence pairs is less than 5.
3. The absolute difference of length between two candidate sentences is more than 7.
4. The number of lexical overlapped items after removing stop-words is less than 3.

The first criterion is adapted from the MSRPC and the third one is similar to the MSRPC's word-based length difference of sentences, which is defined as 66.6%. Although the lexical overlap chosen in the MSRPC is 4, given the nature of the text (which is relatively high in overlapped items due to named entities and lengthy sentences of stop words), we found that the numbers used in the MSRPC are not useful for the data we collected. On the other hand, in Twitter, tweets of a length less than 3 are filtered out and the remaining candidate pairs are pruned if the Jaccard Similarity score of the pair is less than 0.5. Considering our data, the filtering criteria that we applied are comparable to MSRPC due to the similarity of the source data.

5.5.1 Extracting Sentences for Ideal Candidate Pairs

While unwanted text is pruned out with the criteria (1 to 3), as previously explained, potential sentences are paired up. To make the selection process of candidate paraphrases easier, we filtered the sentence pairs with at least 3 overlapping words after removing stop words (criterion 4). Each pair is stored with a list of lexical overlap words and their attributes. For instance, a pair of sentences is shown below (Figure 5-3) with its attributes.

```
['ikramiye', 'Kurban', 'maaş', 'emekliye']
5       9       11

"Ramazan ve Kurban bayramlarında emekliye bir maaş ikramiye vereceğiz.

Sanıyorlar ki emekliye Ramazan ve Kurban Bayramı'nda birer maaş ikramiye verdik.
```

Figure 5-3: Representation of candidate sentence pair

The words in the list are the overlapping words of the candidate sentence pair. The numbers 5,9, and 11 are the word-based number of lexical overlapped items, length of source sentence, and length of target sentence, respectively. The list above shows only four overlapped words between the two sentences, while the number of overlapping words shows five. That is because the list excludes any overlapping stop words. In Figure 5-3, the stop word *"ve"* is excluded from the list. In this way, we can easily identify sentence pairs to decide whether they have any common content word that means they are a potential paraphrase pair. The "stop word list" of Turkish words used in our experiment was

obtained from a Lucene project[15], and is an extended version of the list provided by Can et al. (2008). The list can be found in Appendix C. Content words increase the possibility of a pair becoming a candidate paraphrase pair, but they are not necessarily paraphrases. For instance, the sample pair above has four overlapping content words, and is hence selected as a candidate pair, but it has been labelled as a negative paraphrase pair. Paired sentences that were identical other than minor differences such as punctuation, named entities etc. were discarded.

Each sentence was paired with every other sentence within the rules we defined earlier. A huge collection of candidate paraphrase pairs, approximately 5,000, were obtained on a daily basis, and were now ready to refine. Each sentence was treated as both a source and target sentence, to find out the best possible match. Once a sentence was matched to another one as a possible candidate pair, both sentences were excluded from being matched to another sentence. Note that using the same source sentence multiple times with different target sentences, as for the Twitter Paraphrase Corpus, might not be appropriate for some techniques. This way, each sentence is used only once either as a source or target sentence. For instance, in Figure 5-4, a source sentence, *"7 Haziran seçimlerinin ardından 25. dönemde görev yapacak milletvekilleri TBMM Genel Kurulunda yemin edecek*.", is paired with four different target sentences. After selecting one of the target sentences, in this case, the sentence *"7 Haziran'da seçilen milletvekilleri bugün Meclis Genel Kurulu'nda yemin edecek."* these two selected sentences are not used with any other pair, either as a source or as a target sentence. As a result of this strategy, our data was more diversified and the possibility of covering a broad range of paraphrase structures increased.

[15]http://alvinalexander.com/java/jwarehouse/lucene/contrib/analyzers/common/src/resources/org/apache/lucene/analysis/tr/stopwords.txt.shtml

```
['edecek.', 'milletvekilleri', 'Genel', '7', 'yemin']
5   14  10

7 Haziran seçimlerinin ardından 25. dönemde görev yapacak milletvekilleri TBMM Genel Kurulunda yemin edecek.

7 Haziran'da seçilen milletvekilleri bugün Meclis Genel Kurulu'nda yemin edecek.

['TBMM', 'yemin', 'milletvekilleri', '25.', 'Genel']
5   14  21

7 Haziran seçimlerinin ardından 25. dönemde görev yapacak milletvekilleri TBMM Genel Kurulunda yemin edecek.

CHP Genel Başkanı Kemal Kılıçdaroğlu, bugün yapılacak yemin töreni öncesi TBMM 25. Dönem CHP milletvekilleri ile Genel Merkez'de b

['TBMM', 'yemin', 'milletvekilleri', '25.', 'Genel']
5   14  21

7 Haziran seçimlerinin ardından 25. dönemde görev yapacak milletvekilleri TBMM Genel Kurulunda yemin edecek.

CHP Genel Başkanı Kemal Kılıçdaroğlu, yarın yapılacak yemin töreni öncesi TBMM 25. Dönem CHP milletvekilleri ile Genel Merkez'de b

['TBMM', 'yemin', 'milletvekilleri', '25.', 'Genel']
5   14  18

7 Haziran seçimlerinin ardından 25. dönemde görev yapacak milletvekilleri TBMM Genel Kurulunda yemin edecek.

Kemal Kılıçdaroğlu, bugün yapılacak yemin töreni öncesi TBMM 25. Dönem CHP milletvekilleri ile Genel Merkez'de bir araya geldi.
```

Figure 5-4: Source and target sentences for choosing a candidate sentence pair

The filtering process is enough to exclude pairs that are unfavourable to be selected as candidates. Even though a high threshold score is set compared to the MSRPC, for common word occurrences we obtained a great number of candidate pairs. This was due to the length of the sentences, stop words, and named entities. We therefore handpicked candidate sentence pairs and ended up with 1,000 candidate paraphrase pairs.

Our corpus is relatively small, but provides a diverse set of examples of the Turkish language, including a variety of texts such as questions, dialogue and so on. Furthermore, paraphrase corpora collection might be a challenging task in terms of finding negative instances, because negative paraphrase pairs do not occur naturally. The MSRPC is constructed from daily news sites, but it does not have natural negative instances, as its pairing process only matches headlines. On the other hand, the Twitter Paraphrase Corpus acquires sentences by content information (topics are used) and matches sentences under the same topic; then annotators judge the similarity of a chosen sentence to multiple other sentences in that topic. Following the strategy used in the TPC, we extracted text and paragraphs under each headline. A paragraph is a more expanded form of text that focuses on different aspects from the headline. It can be stated that a headline is the core of a paragraph and each sentence within this paragraph circles around the headline within a

different distance. Therefore, it is easier to obtain negative paraphrases from a paragraph of sentences that almost never mean the same but have the same context.

5.5.2 Drawing Inferences from the Process of Corpus Construction

During this work, we observed that news web-page titles along with headlines are more suitable for the construction of a Recognizing Textual Entailment corpus, which requires a unidirectional relationship only. This is because a title contains short and limited information pointing out a story that is associated with its headline, which is expanded with a little more information. An example title-headline is stated in Table 5-1 along with its translation.

Title	CENAZE TOPRAĞA VERİLDİKTEN SONRA BEACH CLUP DA EĞLENCE
Headline	Cenazenin toprağa verilmesinden sonra Kuşadası Karaova sahilinde, Wilkinson'un çok sevdiği Emyr Beach Clup'ta bir parti düzenlendi.
Title	BEACH CLUB ENTERTAINMENT AFTER THE FUNERAL
Headline	After giving the funeral of land in Kusadasi Karaova coast, Wilkinson held a party at his beloved Emyr Beach Club

Table 5-1: An example that shows relations between titles along with its headline in both Turkish and translation to English

One other important aspect is the identification of natural negative paraphrases. While the definition of paraphrase is still imprecise, it is not possible to define what is not a paraphrase in terms of its rules or meaning. This problem is also pointed out, and the need for a specific paraphrase definition addressed, by Rus et al. (2014). However, one way of obtaining natural paraphrases might be to relate them to how they are semantically similar. Degree of semantic similarity of two sentences shows whether these two sentences have a bidirectional or unidirectional relation, or have no relation at all. This issue has been tackled in the TPC, by converting the scores of annotators to a semantic similarity score, 0 to 5, for each of the pairs. This does, however, lead to the issue of deciding which score belongs to a paraphrase and which to a non-paraphrase category.

5.6 Annotation and Human Agreement

We followed the same scoring methodology used in the TPC, assigning a similarity score for each paraphrase pair. We asked eight native Turkish speakers to give a score between 0 and 5 to each pair according to their semantic equivalency. The data was split into two parts (1,002 sentence pairs in total) and four different annotators judged each part. The criteria for scoring were based on the Sentential Semantic Similarity task using Rus, Lintean, Banjade, Niraula, & Stefanescu's (2013) methodology, which will be explained later.

In addition, collected data is not lowercased; numbers or named entities are not replaced with generic names, unlike in the MSRPC, and there is no pre-processing applied.

5.6.1 Preparation for Annotation

Before handing the sentences to annotators, a small preliminary experiment was completed to clarify the task, as the word "paraphrase" does not directly translate to Turkish, so it needed elucidation.

First, we prepared a form consisting of two short videos (Appendix C). These two videos were adapted from the Microsoft Research (MSR) Video Description Corpus (Chen & Dolan, 2011). As mentioned in Section 5.2, the objective of constructing this corpus was to generate a paraphrase evaluation corpus by asking multilingual Mechanical Turk workers to describe the videos. We hypothesised that confronting our annotators with the way other annotators described the same videos is an effective approach to familiarising them with the task. They summarised the videos with one sentence and, afterwards, we gathered all annotators' descriptions for those videos. Then we showed them how their interpretation could be different from others. The interpretation or wording for the same video is different so that they can see the others' points of view. Our annotators responded that the way the paraphrasing task is explained in the guidelines is not really as clear as practising the situation.

Video description sentences from the annotators were collected and included in "Guidelines"[16], which were given to them beforehand. The guidelines also include an example pair for each rating score from 0 to 5, along with a short explanation. These

[16] The guidelines can be found in Appendix C along with their original version.

examples were chosen from multiple samples by asking three different native speakers who gave exactly the same score for those instances. After completing this preliminary experiment, sentence pairs were sent to each annotator. Our annotators commented that this small task gave them a better understanding of how to conduct the scoring.

5.6.2 Annotation Guidelines

The gold standard paraphrase corpus, the MSRPC, is supplied with a binary judgement: with positive (1) or negative (0) pairs. As there is still debate as to whether a binary judgment or fine-grained scoring better satisfies a paraphrase corpus, one of the latest paraphrase corpora, the TPC, is supplied with a fine-grained scoring scheme converting the number of an annotator's answer to binary scoring. With this approach, the data is made viable for paraphrase identification and sentential semantic similarity.

Given the inadequacy of binary judgement with regard to paraphrasing, we chose to provide data of sentential textual similarity task by agreeing on fine-grained scoring, knowing that it could be converted to binary, but not the other way around. We followed the guidelines provided for the semantic similarity task (Agirre et al., 2012) and annotators scored sentence pairs with the criteria shown in Table 5-2.

5- IDENTICAL	Completely equivalent; they mean the same thing.
4-CLOSE	Mostly equivalent, but some unimportant details differ.
3- RELATED	Roughly equivalent, but some important information differs/missing.
2 - CONTEXT	Not equivalent, but share some details.
1-SOMEWHAT RELATED	Not equivalent, but are on the same topic.
0- UNRELATED	On different topics

Table 5-2: Sentential Semantic Similarity scores for candidate paraphrase pairs

As mentioned, these similarity scores were introduced to the annotators, along with an example of each scoring scheme, in the guidelines. We did not give them any time restriction on the task because they mentioned that some pairs were confusing meaning scoring required more time.

For the purpose of the paraphrase identification task, the scores were converted to a binary judgement. Firstly, the answers of each annotator were converted to a binary judgement by accepting the sentence pairs marked as identical (5), close (4) and related (3)

as positive pairs (1) and the other ones, context (2), somewhat related (1) and unrelated (0) as negative pairs. The number of positive and negative answers for every instance is demonstrated by tuples such as (1,3), which shows that 3 annotators out of 4 judged a pair as a negative paraphrase and only one of the annotators judged it as a positive paraphrase. In Table 5-3, we show the criteria of the binary judgment based on the number of annotators' answers.

Number of answers	Judgement
(4,0); (3,1)	Positive (1)
(0,4); (1,3)	Negative (0)
(2,2)	Debatable

Table 5-3: The criteria of binary judgement based on the number of annotators' answers

We generalise the semantic similarity score with an aggregated approach, which helps to identify debatable pairs. This approach is similar to the TPC labelling method, which is also based on agreement between five annotators. In the TPC, debatable pairs are represented by the tuple (3,2), meaning that a sentence pair is accepted positive by three annotators and negative by the other two. Positive and negative pairs are represented by tuples (5,0), (4,1) and (2,3), (1,4), (0,5), respectively.

For our semantic similarity task, we also provided the average score, which is calculated based on the given scores of annotators along with the tuples. The average scores range between 1.75 and 3.00 for debatable pairs, whereas positive pairs are higher than 3.00 and negative pairs are scored as less than 1.75. Additionally, the criteria defined in Table 5-3 can be interpreted in a range between 0 and 1 that follows: (4,0): 1.0; (3,1): 0.75; (2,2): 0.50; (1,3): 0.25 and (0,4): 0.0.

5.6.3 Inter-Annotator Agreement

The scoring scheme was a perplexing problem for our annotators. They said that the difficulty of scoring is not in deciding whether sentences are similar, but deciding the degree of similarity or dissimilarity.

The ratings from each annotator were analysed. After analysing the data, there were some inconsistencies detected between annotators' answers, as expected. We applied

Cohen's Kappa (Cohen, 1960) to see the agreement between each pair of annotators on each part of the dataset. In general, a Kappa coefficient value of 1 indicates strong agreement between annotators, whereas a result of 0 shows an agreement by chance. A negative value (<0) indicates no agreement. However, the interpretation of the interval values has caused disagreement between researchers (Landis and Koch, 1977; Fleiss, 1981; Altman, 1991 cited from Gwet, 2012). A task-oriented approach is accepted as one that depends on the type of data to be analysed. Paraphrasing tasks rely heavily on subjective inferences, so we used Landis and Koch's (Landis and Koch, 1977 cited from (Gwet, 2012)) agreement interpretation scores, see Figure 5-5.

Kappa value	Strength of agreement
< .20:	Slight
.21-.40	Fair
.41-.60	Moderate
.61-.80	Substantial
0.81-1.0	Almost perfect

Figure 5-5: Landis and Koch (1977) interpretation of inter-reliability scores

Table 5-4 shows the kappa results between every two annotators. The annotators are identified as R1, R2, R3 and R4 and although the notation for the annotators is the same for two parts of the dataset, four different people annotated each part. Because of this, we first report the individual results of each part and later recalculate the agreement on the whole data after concatenating the two datasets.

Annotators	**First 500 pairs**	**Second 502 pairs**
R1-R2	0.47	0.42
R1-R3	0.56	0.46
R1-R4	0.56	0.37
R2-R3	0.40	0.31
R2-R4	0.55	0.23
R3-R4	0.50	0.33

Table 5-4: Cohen's Kappa results obtained on individual parts of data for inter-agreement of annotators

The highest kappa value obtained was of 0.56, obtained between three annotators, R1, R3 and R4, on the first half of the data. This result is interpreted as "moderate

agreement" according to Landis and Koch (1977) (cited from (Gwet, 2012)) The lowest agreement (0.23 kappa value) occured between R2 and R4 on the second part of the data. This result is interpreted as a fair agreement. Cohen's Kappa can be used only between two annotators and these results do not reflect the inter-agreement reliability of the full dataset, but they are given for informative purposes only. To compute the inter-agreement of more than two annotators, we use another statistical approach: Fleiss's Kappa (Fleiss, 1971) shows the inter-reliability of multiple annotators. We obtained the whole data by concatenating the individual parts. Different annotators marked every individual part of the data. This raises the question "Which two annotators' results are selected to concatenate?" Although concatenation of the two parts does not affect the individually marked sentences pairs, the inter-reliability agreement of annotators might change Cohen's Kappa. However, the Fleiss Kappa results will not change, due to its instance-based formula, which mainly relies on the number of scores given to each instance. Therefore, we calculated the Fleiss Kappa score on the full dataset using two different judgment processes: semantic equivalency and binary judgment. As previously mentioned, scores that are based on semantic similarity can be converted to a binary judgement for the purpose of paraphrase identification tasks.

Scoring	Fleiss Kappa (%) on Whole data
Degree of Semantic Equivalency (0-5)	0.17
Binary Judgment (1,0,Debatable)	0.42

Table 5-5: Fleiss Kappa score is computed based on the two different judgment criteria

Table 5-5 shows the results of inter-agreement of annotators on the full dataset. The results show a slight agreement with regards to semantic equivalency. The inter-agreement increases with the binary judgment as it is expected. Binary judgment can bee seen as a more coarse-grained measure of semantic equivalence. A higher inter-agreement with binary judgment shows that there is a correlation between annotator's answers. They tend to give close results that can only be unified by narrowing the judgement criteria.

5.6.4 Annotator Bias

There are different factors that affect the ratings between annotators. The fact that paraphrasing is a subjective interpretation means it is unlikely that a very high agreement

will be reached for all sentence pairs in terms of paraphrasing. Other factors, such as an annotator's proficiency, language competence, and inferential skills may explain the variety of scoring.

As well as the inter-reliability scores, there is a bias index given for each sentence pair's scores. There were 23 sentence pairs detected that were judged as identical (5) by at least one of the annotators, but judged by at least one as unrelated (0). Apart from these 23 sentence pairs, the given scores are most likely to be in close range for the same sentence pairs. For instance, a sentence pair may be rated as 4 by one annotator, while the other annotator may rate it as 3. However, it is hard to draw a line owing to the complexity of fine-grained scoring scheme.

5.7 Turkish Paraphrase Corpus (TuPC)[17]

Our newly constructed paraphrase corpus finally consists of 1,002 sentence pairs, which are labelled for both tasks: sentential semantic similarity and paraphrase identification (Eyecioglu & Keller, 2016) .

After conversion of the scores into binary, we obtained 563 positive, 285 negative, and 154 debatable pairs. As a result of excluding the 154 debatable pairs, the TuPC has 848 sentence pairs that can be used for paraphrase identification tasks. Detailed data statistics of TuPC regarding the agreement between four annotators is shown below, in Table 5-6.

	Agreement	Number of pairs	Value
Positive	(4,0)	376	563
	(3,1)	187	
Debatable	(2,2)	154	154
Negative	(1,3)	151	285
	(0,4)	134	

Table 5-6: TuPC data statistics: positive, negative and debatable sentence pairs

5.7.1 Train and Test Set

There are various ways to split a dataset into train and test sets. However, there are no obvious criteria described for splitting a paraphrase corpus into train and test sets. The

[17] In order to download TuPC: https://osf.io/wp83a/

percentages of train and test sets in the paraphrase corpora that we experimented with are varied: the TPC has relatively a small test set compared to its training set (838 and 11,530 sentence pairs, respectively, after removing debatable pairs). The percentage of the train to test sets in the MSRPC is 70.3% to 29.7%. The PAN Corpus has approximately 77% of its sentence pairs in train set and 23% in its test set. Hence, the percentage of sentence pairs (approx.) selected for the TuPC was 60% for the train set and 40% for test set. Therefore, we obtained a train set of 500 pairs and a test set of 348 pairs. The TuPC contains 339 positive and 161 negative sentence pairs in the train set and 224 positive and 124 negative pairs in the test set.

Note that the TuPC was shuffled randomly before splitting the data into the train and test sets.

5.7.2 Baseline

We established a simple baseline by labelling every sentence pair as a positive paraphrase.

The proportion of positive paraphrases is larger than the proportion of negative paraphrases in the dataset. Therefore, the F-score is quite high, whereas the accuracy is lower, like the baseline of the MSRPC. The baseline results are presented in Table 5-7.

Baseline
Accuracy= 0.6639 Precision= 0.6639 Recall = 1.0 F_score= 0.798

Table 5-7: Baseline of TuPC computed for paraphrase identification task

5.8 Paraphrase Identification on Turkish Paraphrase Corpus

One objective of this chapter is to learn whether knowledge-lean methods are applicable to a language other than English. The methods we experimented with in Chapter 4 are applied to the TuPC along with methods that tried in Chapter 3. All experiments were completed on the dataset after the debatable pairs had been removed.

5.8.1 Pre-processing and Simple Similarity Measures

The raw data includes punctuation and spelling errors that have not been hand-corrected. We performed tokenisation and lowercasing on the TuPC. Splitting the sentences at white

spaces performs tokenisation, as we want to keep pre-processing to a minimum. Lowercasing is performed for a fair comparison to other corpora that we experimented with.

In Chapter 3, we explored four different similarity measures using simple overlap features of the MSRPC and PAN. These measures provide an overall structure of the dataset by performing on overlapped items. The same classification methodology is applied to the TuPC.

5.8.2 Extracting Character and Word N-gram Features

In Chapter 4, the best results on the Twitter Paraphrase Corpus are obtained using simple character and word n-gram features, and with their combinations. The set of four features (U, N, L1 and L2) are extracted from the TuPC to be used with SVM classifiers. The same notation is used for the TuPC results so that character bigrams are denoted as C2, and C2W1 shows the combination of character bigram and word unigram features.

5.8.3 Results

This section presents the TuPC results using the methods applied in Chapter 3 and Chapter 4. We then examine the TuPC results considering the previous paraphrase corpora results obtained from same methods.

5.8.3.1 Similarity Measure Results

Table 5-8 presents the simple overlap results from the TuPC computing four similarity measures.

	Character Bigrams				Lexical			
Measures	**Acc.**	**Pre.**	**Rec.**	**F-sc.**	**Acc.**	**Pre.**	**Rec.**	**F-sc.**
Dice	72.7	74.0	88.8	80.7	75.0	79.7	82.1	80.9
Cosine	69.0	71.8	85.3	78.0	76.2	79.3	85.3	82.2
Jaccard	73.3	74.9	87.9	80.9	75.3	79.5	83.0	81.2
Jaccard_Gen	73.6	76.8	84.4	80.4	75.3	79.0	83.9	81.4

Table 5-8: Similarity measures results of TuPC on character and word level

We obtained these results (Table 5-8) by looking at the threshold on the train set; the threshold is tuned for each similarity measures separately. The highest results are obtained with the Cosine measure using lexical features, in contrast to using character level

features, which gives the lowest results in overall. We hypothesize that this is because of the highly agglutinative nature of Turkish, in which words are formed by variations of unchanging roots, causing misidentification of pairs in terms of character bigrams. In particular, common morphological features, which form a large part of the character grams, tend to cause misidentification of non-paraphrases, leading to lower precision for character bigrams.

In general, word-level features seem to perform better than character-level features.

5.8.3.2 SVM classification and Results

Cross validation may be the better choice in a case where the data is limited. We applied 10-fold cross validation with SVM, as we already experimented the same classification method for the MSRPC and PAN in Chapter 4 (Section 4.6.2). Table 5-9 demonstrates the results of the TuPC with two different kernels of SVMs.

SVM (Linear kernel)					**SVM (RBF kernel)**				
Features	**Acc.**	**Pre.**	**Rec.**	**F-sc.**	**Features**	**Acc.**	**Pre.**	**Rec.**	**F-sc.**
C1	66.4	66.4	100.0	79.8	C1	68.8	68.7	97.3	80.5
C2	77.5	80.8	86.9	83.7	C2	76.4	78.8	88.3	83.3
W1	73.6	75.5	89.3	81.8	W1	72.2	75.4	86.5	80.5
W2	71.2	72.7	90.9	80.8	W2	71.3	73.4	89.2	80.5
C1C2	76.8	80.5	86.0	83.1	C1C2	75.5	77.6	88.8	82.8
W1W2	73.7	76.1	88.5	81.7	W1W2	73.7	76.7	86.9	81.4
C1W1	72.6	75.1	88.1	81.0	C1W1	72.4	74.7	88.6	81.0
C2W2	76.7	80.3	86.1	83.1	C2W2	76.4	78.7	88.5	83.3
C1W2	71.1	73.2	89.4	80.4	C1W2	71.6	72.9	91.3	81.0
C2W1	76.4	79.6	86.9	83.0	C2W1	76.3	78.7	88.3	83.2
C12W12	77.5	81.0	86.5	83.6	C12W12	74.5	77.0	88.1	82.1

Table 5-9: TuPC results obtained from character and word n-gram features of SVM (Linear and RBF kernels)

The highest result is obtained on character bigrams (C2) with the linear kernel. Character bigrams with the RBF kernel is slightly lower, but is still the highest second result in the overall table. It seems that combinations of all features (C12W12) do not add any performance over character bigram (C2), either for linear or RBF kernels. In fact, the results from the combined features with the linear kernel are lower than that of the individual features (C1, C2, W1, W2) results. With RBF kernels, the combinations of word unigrams and bigrams (W1W2) perform well compared to their individual results (W1 and W2).

The overall table (Table 5-9) show that the obtained results are competitive in comparison to the baseline results (*accuracy=66.4 and F-score=79.8*). The linear kernel is not able to classify paraphrase pairs correctly using C1 features only.

Table 5-10 presents the best results obtained on TuPC. The linear kernel is notably high compared to the Cosine measure. Considering the size of the TuPC, the applied methods perform quite well in comparison to the baseline.

Model	Acc.	F-sc.
Cosine (word_level)	76.2	82.2
C2 (Linear Kernel)	77.5	83.7
Baseline	66.4	79.8

Table 5-10: The best results on the TuPC

5.8.3.3 Comparison to other paraphrase corpora

Referring to the results on the MSRPC (Chapter 4), we can infer that character bigrams are the best features for the TuPC, as well as for other paraphrase corpora. One of the reasons is sentences are constituted of space-separated words and follow a word order rule, even though the word ordering differs between the languages (*Subject-Verb-Object* in English; *Subject-Object-Verb* in Turkish). Examining the TuPC for character bigrams with Dice Coefficient; the error rate is 0,27 and false positive rate is 0,56. The same measure with the same set of features used in MSRPC resulted in false positive rate of 0,54 (See Chapter 3-Section 3.6).

5.9 Discussion

As mentioned in Chapter 3, pre-processing of the raw data plays an important role in that the results from the same methods but using different pre-processing tools might differ significantly. Hence, Rus et al. (2014) strongly suggest that paraphrase data should come with some pre-processing to enable a fair comparison of applied methods. Therefore, we aim to release different variants of pre-processed data (tokenised, PoS-tagged) soon, and make it available to the research community.

Annotation is a very laborious process, which impedes us in constructing a larger corpus, although we could have obtained abundant data within a short time. The crowdsourcing methodology overcomes this problem for the English language, but

crowdsourcing is not widely used yet for Turkish language. Moreover, choosing between binary judgment and fine-grained scoring is a trade-off when it comes to corpora construction. One reduces the laborious work with a shallow rating scheme, and the other takes into consideration even minor differences within a limited dataset.

Although the quality of a paraphrase corpus is improved by removing debatable cases, it is worth investigating how systems perform on such cases, particularly for systems that are only trained on the clear cases.

One significant outcome from using cosine measures should be pointed out. Cosine measures behave notably differently on each paraphrase corpus, whereas other measures (Dice, Jaccard, Jaccard Generalization) showed comparable results across the corpora.

5.10 Overall Summary

In this chapter, a combined way of collecting data taking account of previous paraphrase construction methods was employed for the construction of paraphrase and semantic similarity data in Turkish. In order to compare the applicability of knowledge-lean methods on a paraphrase corpora not in English, the TuPC was used to experiment with knowledge-lean techniques that were originally developed for the other paraphrase corpora: the MSRPC, TPC and PAN, and the results obtained from the TuPC are compared to those paraphrase corpora results.

Moreover, the paraphrase corpora in other languages are reviewed pointing out their limitations for paraphrase identification tasks. We draw some attention to the possible tasks of textual entailment and summarisation relative to paraphrasing identification tasks.

The results obtained from similarity measures are lower as compared to the results obtained from SVMs using a set of four features. One observation from the results of TPC and TuPC is the apparent similarity of the experimental results obtained using the same set of features.

As far as we are aware, the Turkish Paraphrase Corpus is unique in terms of providing both negative and positive instances, and is currently the only available paraphrase corpus for paraphrase identification and semantic textual similarity in Turkish.

The TuPC is relatively small compared to other paraphrase corpora, but provides a diverse set of examples of the Turkish language, including a variety of text types.

In the next chapter, a knowledge-lean approach using continuous representations of words and characters is presented for identifying semantic relations between a pair of sentences.

Chapter 6

6 Distributed Sentence Representation with Vectors of Words and Characters

6.1 Introduction

The previous chapters have shown that knowledge-lean methods can be applied successfully for paraphrase identification. One key challenge of these methods is whether semantic knowledge can be obtained using text-based statistics, which then can be used for detecting paraphrases without relying on tools for semantic and syntactic analysis.

Mikolov, Corrado, et al. (2013) introduced two new architectures of Neural Networks: continuous-bag-of-words (CBOW) and skip grams (SG). With these architectures, words are represented as vectors according to their co-occurrence with other words. Although these Neural Network (NN) models share the same theory of distributional similarity models (DSMs), there are differences that make NN models more popular. Baroni, Dinu, and Kruszewski, (2014) performed various experiment on different tasks in order to compare the performance of these two models: NN models are defined as *predict models* and the DSMs as *count models*. The DSMs rely on raw co-occurrence counts of words and then a variety of techniques (e.g. dimensionality reduction such as SVD and weighting) are applied to obtain more informative vectors. In contrast, with NN models similar word vectors are directly predicted according to their context and as a result computation cost is remarkably reduced.

These Neural Network (NN) models have been shown to perform well on the development of recent PI methods (H. He et al., 2015; Socher et al., 2011; Yin & Schütze, 2015; Zarrella et al., 2015; J. Zhao & Lan, 2015). A growing body of research has recently

been conducted on paraphrase identification using a variety of NN Models, the main focus being representing long-text fragments. While that research focuses on larger fragments of text, such as phrases, sentences and even paragraphs, an alternative approach is presented in this chapter by introducing vectors of character n-grams that carry the same attributes as word vectors. Previous work has shown that the proposed method has the ability to capture syntactic relations as well as semantic relations using simple vector operations (Mikolov, Chen, Corrado, & Dean, 2013; Mikolov, Corrado, et al., 2013). This chapter shows that NN approach can be competitive on Twitter compared to more sophisticated methods.

Although CBOW and SG models in Neural Networks are often referred to as "deep learning" concept, the word vectors are simply obtained from a shallow word embedding process, and we consider them knowledge-lean. One of the reasons for this is that the word vectors are produced from unlabelled data by simple usage of frequency of word occurrences. In addition, the obtained word vector encodes semantic information according to its relative importance to the data, without the usage of any external tool or annotated text. Moreover, these are log-linear models, which reduce the complexity and cost of training a very large dataset, by making the training process much faster. These NN methods, also called *"word2vec"*, are of interest to PI tasks, where two text fragments are paired mainly based on lexical overlap features.

This chapter proceeds with a brief introduction to recent research; CBOW and SG models and the attributes used to construct those models that are used in our experiments. The word embedding strategy, with words and also characters, will be explained. The training data selection will be explained in order to obtain word and character vectors from the constructed models using the experimental paraphrase corpora: the TPC and the MSRPC. Paraphrase Identification tasks work on sentence pairs and this raises the question of how a sentence or sentence pair should be represented with word vectors. The possibilities arise with this question of whether a sentence pair should be represented as one vector or each sentence in a pair should be represented, after doing some computations between individual word vectors. Although recent research has taken steps in extending these word vectors to phrases, sentences and documents (Le & Mikolov, 2014), we introduce an alternative approach by building models from character n-grams and

constructing character vectors in the same way words are used. Hence, considering the approaches presented in the literature, a variety of vector operations has been tried with word and character vectors, and these are then turned into simple features of SVM classifiers. Later, the features that are based on NN models are investigated independently, by adjoining them to the previous features experimented with in Chapter 4. We conclude with a comparison of the results from the models and the features with respect to the SVM kernels. Our experiments show that there is a wide range of possible future experiments that present themselves regarding the usage of word2vec, and we draw attention to the significant ones.

6.2 Neural Network Architectures: CBOW and SG

Distributed representations of words using continuous bag-of-words (CBOW) and skip-gram (SG) models were first introduced by Mikolov, Corrado, et al. (2013).

The implementation of these two algorithms differs in terms of their neural network layers. As shown in Figure 6-1, CBOW models accept a word within a window size, and project word vectors from the distributed properties of words that are observed from training data in the input layer. The word is mapped onto a high dimensional vector in the hidden layer, and then the word vector is presented in the output layer. SG models can be seen as a reversed process of the input and output layers of CBOW. In SG models, a word is accepted in the input layer to predict the words by skipping the words within a certain range in the output layer. The current word is again mapped into a high dimensional vector in the hidden layer, then the output layer represents the predicted neighbouring words of the given word within a window size.

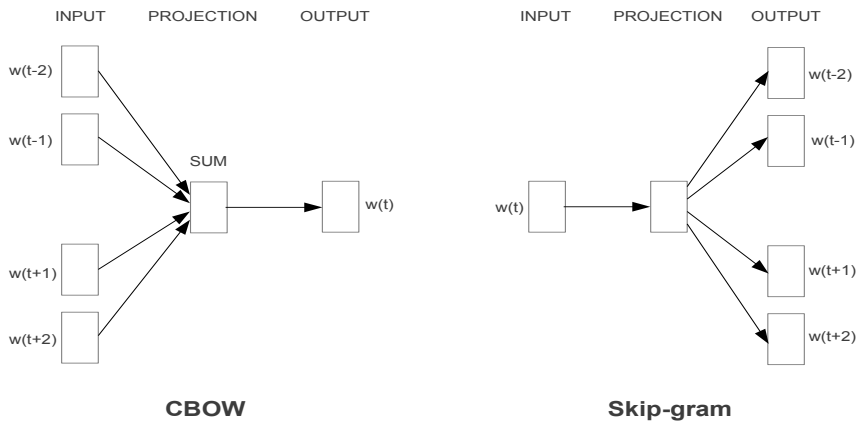

Figure 6-1: Representation of CBOW and SG models (Mikolov, Corrado, et al., 2013)

CBOW models differ from standard bag-of-words models by having a continuous representation of the words. SG models can be seen as input-output reversed models of CBOW. One of the reasons that these models have attracted interest in the NLP community is that the computational complexity during the training process is reduced efficiently. It is stated (Mikolov, Corrado, et al., 2013) that the reduced computational complexity even makes it possible to train these models with an unlimited size of vocabulary.

6.2.1 Word and Character Embeddings

Sentence representation is important for PI tasks. Each sentence in a pair is represented by a word vector. As an example of these two models, an actual sentence from TPC is shown below:

"*peter pan is coming on abc family right now*"

CBOW	*{peter pan, pan is, is coming, coming on, ...}*
SG	*{peter is, peter coming, pan coming, pan on, is on, is abc, ...}*

Table 6-1: An actual sentence representation of CBOW and SG models

In Table 6-1, the CBOW model shows each word of the sentence with its surrounding words in a window size of 2. The SG model requires the number of skipped words to be specified, within the specified window size. We have shown a skip gram of 3 within the window size of 2.

In most NLP methods, it is considered that the meaningful and atomic fragments of context-based text are the words. Linguistically, there are smaller units than words that may be considered to have semantic content, such as morphemes, lemmas or stems. On the other hand, fragments such as syllables may not be considered to have any intrinsic meaning, but may be structurally important. Regarding PI, where a semantic equivalency of two sentences is measured within the given context, dividing the texts into sub-word features may have advantages. On the other hand, there is some cost involved in identifying such linguistically relevant features. However, character n-grams might implicitly encode useful information about the multiple kinds of syllables, stems and lemmas of a word. Therefore, our knowledge-lean methodology utilises *character trigrams*, as well as word embeddings for sentence representations.

In Chapter 4, we mainly focused on character bigrams with overlap features. In the work reported in this chapter, our experimentation on overlap methods extended to trigrams, and the obtained results from trigrams show an improvement over bigrams; this is to say that trigrams of characters are more informative than that of bigrams and unigrams. Additionally, representing a sentence in terms of adjacent character trigrams performs a similar function to tokenisation to form words.

The sentence in Table 6-2 shows how a sentence is represented with adjacent trigrams in order to build character vectors. Trigram tokens are treated like words during the process of building statistical models. The word-separating spaces are shown with "␣".

Original sentence	*peter pan is coming on abc family right now*
Character trigrams of sentence	*pet ete ter er␣ r␣p ␣pa pan an␣ n␣i ␣is is␣ s␣c ␣co com omi min ing ng␣ g␣o ␣on on␣ n␣a ␣ab abc bc␣ c␣f ␣fa fam ami mil ily ly␣ y␣r ␣ri rig igh ght ht␣ t␣n ␣no now ow\n w\n*

Table 6-2: Sentence representation; sentence is split into adjacent trigrams

A brief overview of the two statistical models used for our experimentation is provided. The difference in the word embedding strategies of CBOW and SG models are explained in general. The same strategy of word embedding is proposed for character n-

grams, and n is chosen to be 3, logically and in accordance with the previous results of Chapter 4.

In the following, the statistical models will be described in relation to the training datasets that are used to construct them. A variety of vector operations on sentences are then provided, with usage of word and character trigram vectors obtained from previously-constructed models. A variety of operations between vectors are used to represent the sentences; the vectors of sentences are then computed with cosine similarity. The results are used as an input feature in combination with the set of four features in the SVM classifiers. The development process of our methods is shown in Figure 6-2.

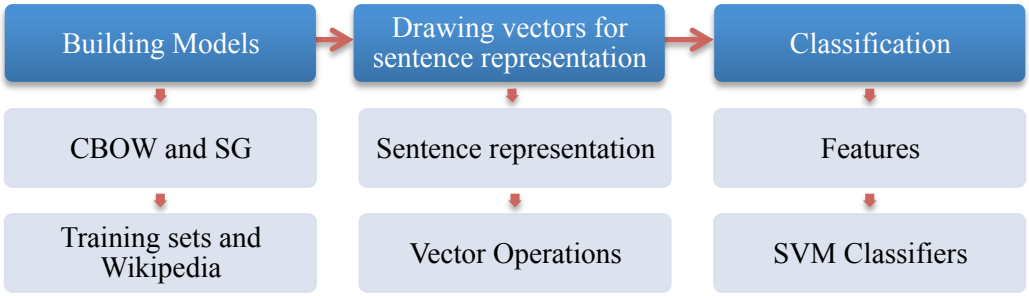

Figure 6-2: The stages of our development process

6.3 Constructing Statistical Models from unlabelled datasets

An important aspect in building statistical models is to decide on the training data. The widespread approach is that the larger the data, the better the results. The quality of training data and the algorithm used to train statistical models are other factors that affect the performance of CBOW and SG models. We first describe the selected training data (Section 6.3.1) and the construction of a variety of statistical models with different parameters (Section 6.3.2) that are trained on those selected datasets.

We use the Gensim[18] package (Rehurek & Sojka, 2010) to build the continuous-bag-of-words and skip gram models used in all experiments.

[18] https://github.com/piskvorky/gensim/

6.3.1 Datasets used for training

There are various decisions to make when constructing a statistical model. The first stage is to decide on a dataset that will be used for training from unlabelled data. This follows the describing the training datasets and constructed models with their attributes that are specific to each dataset. Next, the training datasets, and constructed models with their attributes that are specific to each dataset, must be described.

The question of whether the corpus itself can be utilised for the cases, where the data is limited, for instance some other languages, is pursued by building statistical models where we experiment only with train sets of corpora: the TPC and the MSRPC. Also, it is stated (Mikolov, Chen, et al., 2013) that SG models work well on small training data, because more instances are created by skipping words, whereas CBOW models require more training data for better accuracy.

Eventually, a large dataset is required to build models, in comparison that needed for models built from training sets of paraphrase corpora. We used a very large set of articles of Wikipedia to train the CBOW models. Additionally, the dataset from Wikipedia articles is used to build models that are used to experiment with each paraphrase corpus.

Datasets*:* Test sets from the MSRPC and the development/test set of the TPC also are split into character trigrams, along with their train sets that were used for building models.

The TPC already comes tokenized and punctuation-free; we only perform lowercasing and normalising of the words with the provided normalisation lexicon (Han & Baldwin, 2011). For a better comparison and suitability of training data and experimental data, the MSRPC is tokenised and lowercased and punctuation is removed.

6.3.2 Training Sets of TPC and MSRPC

As mentioned, the Twitter Paraphrase Corpus has target sentences that are paired from a selection of sentences (Chapter 4), and therefore the training set consists of repetitive sentences. All duplicated sentences have been removed to build a model from a unique set of sentences. The final set includes 11,922 sentences, out of a total of 23,026. The Microsoft Paraphrase Corpus includes many fewer duplicated lines, as expected. Its final set includes 7,816 sentences, out of a total of 8,152.

These training sets are used for building both word-based and character-based statistical models.

6.3.3 Wikipedia Dataset

We used Wikipedia-articles-dump file (*downloaded at 07/07/2015*) with a size of 11.98 GB[19]. In order to perform a knowledge-lean approach, the obtained data is only tokenised, lowercased, and cleared of punctuation. Documents are articles and those that exceed 1,000 words or are shorter than 50 words are removed from the file during the cleaning process.

For word-based models, we use the whole Wikipedia dataset. The Wikipedia dataset that is used to construct character-based models split into trigrams only uses one third of the Wikipedia dataset (approx. 4.29 GB). We obtained two different datasets from Wikipedia; their sizes are shown in Table 6-3.

Statistical Properties	**Wikipedia (words)**	**Wikipedia (trigrams)**
Total lines	3,831,719	108,451
Total words/ trigrams	2,065,994,445	1,068,456,094
Unique words/trigrams	8,179,596	91,686

Table 6-3: Statistical properties of statistical models of words and character trigrams built from Wikipedia dataset

The number of total words is approximately the twice of the number of total character trigrams, which are extracted from the two datasets. However, the unique number of character trigrams is notably smaller than the number of unique words, as expected.

6.3.4 Statistical Models

The performance of a model is affected by its properties, such as the size of word vectors, the number of words to be counted after the target word, and whether it includes the less frequent words from training data. These features should be taken into account in combination with the model's architecture; whether it is a CBOW or a SG. These should be defined before building the statistical models. Although experiments by Mikolov, Yih, & Zweig (2013) gave a clear idea of how to choose the appropriate features for a word similarity task, the same features might not be ideal for PI tasks, because there is not just

[19] We obtained a plain file of documents by cleaning the xml tags from the file. All types of symbol have been removed remaining only words during the cleaning process.

one explicit way to represent longer text fragments from word vectors. Therefore, the properties of the models are chosen in consideration of the size and type of the training data, which will be explained next.

Statistical models that are derived from word and character trigrams are notated as ***w2v*** and ***ng2v***, respectively. The size of a model corresponds to the unique number of tokens (words or character trigrams) in each training dataset. Based on the number of tokens in datasets, each of these vectors is projected in the hidden layer with a specified dimension, which is notated as ***"s"***. There are different vector sizes, from 20 to 600, experimented with in Mikolov, Corrado, et al. (2013)'s work. They suggested that simultaneously increasing the dimensionality of vectors and the size of the training data increases the accuracy on semantic-syntactic word relationship tasks, using the same set of attributes. Hence, two different vector sizes are used in our experiments: 200 and 400, considering we have both small and large training data sets. We have chosen the surrounding words of within a 5-window size ***(ws)*** for each model. In most NLP methods, words that rarely occur in vocabulary are removed by defining a threshold of frequency of their occurrences. Although this attribute might be useful for some NLP tasks, we aim to utilise all vocabulary items, so the minimum count ***(mc)*** of rare items is set to 1. Because training sets are quite small in quantity, for SG models the skip-gram count ***(sg)*** is set to 2.

There are 12 statistical models built in total. Four are built from Wikipedia, (two for ng2v and two for w2v) in order to be used with both the TPC and the MSRPC. The other eight models are built from the training sets of the TPC and the MSRPC, with four different models for each corpus (2 for ng2v and 2 for w2v). As a result, each corpus, the TPC and the MSRPC, is used to experiment with eight different models; four models utilise the training sets and the other four models utilise Wikipedia, using different attributes.

The same attributes are assigned to both models, of word and character vectors. Because CBOW models predict the current word based on the context, and the SG models predict surrounding words given the current word, we assume that small data might be represented well with SG models and CBOW on a larger dataset (Mikolov, Corrado, et al., 2013). Accordingly, the Wikipedia dataset is only trained with CBOW models, whereas the smaller corpus-specific training data has been used to train all models. These models are

demonstrated in Table 6-4: Model 1, Model 2, Model 3 and Model 4 are trained on training sets of paraphrase corpora, while the Wikipedia trained models are denoted with prefix "Wiki", such as WikiModel 1. Among the models constructed from the training set, Model 1 and Model 2 are trained on the SG algorithm for w2v and ng2v, respectively; and Model 3 and Model 4 trained on the CBOW algorithm for ng2v and w2v, respectively. In terms of the size and type of the vectors, they carry the same attributes. The Wikipedia models are only trained on the CBOW algorithm: WikiModel 1, WikiModel 4 for w2v models with 200 and 400 vector sizes, respectively, and WikiModel 2 and WikiModel 3 for ng2v models with 200 and 400 vector sizes, respectively.

MODELS	Training data	Training algorithms	Model attributes	Type of vectors
Model 1 (M1)	Training sets of TPC and MSRPC	SG	s=400;ws=5;mc=1	w2v
Model 2 (M2)			s=400;ws=5;mc=1	ng2v
Model 3 (M3)		CBOW	s=400;ws=5;mc=1	ng2v
Model 4 (M4)			s=400;ws=5;mc=1	w2v
WikiModel 1 (WikiM1)	Wikipedia	CBOW	s=200;ws=5;mc=1	w2v
WikiModel 2 (WikiM2)			s=200;ws=5;mc=1	ng2v
WikiModel 3 (WikiM3)			s=400;ws=5;mc=1	ng2v
WikiModel 4 (WikiM4)			s=400;ws=5;mc=1	w2v

Table 6-4: Constructed models for word and character vectors and their attributes

Next, a few examples are presented from w2v and ng2v models in order to evaluate the accuracy of the models. We have shown only the results of Wikipedia trained models, because training models do not have enough vocabulary for the comparison of the models.

6.3.5 From Words to Vectors

The constructed statistical models are tested with few prototype similarities that are represented in Mikolov's work (Mikolov, Yih, et al., 2013). For instance, the sum of vectors of the words "woman" and "king" is subtracted from the vector of the word "man", which gives the vector of word "queen". In addition, the similarities of words "woman-man" and "teacher-student" have been computed using our w2v models. These results are shown in Table 6-5.

Examples	Models	Similarity Scores
a) $(V_{woman} + V_{king} - V_{man})$ b) $similarity(V_{woman}, V_{man})$ c) $similarity(V_{teacher}, V_{student})$	WikiModel 1	a) queen, 0.6240 b) 0.7435 c) 0.681
	WikiModel 4	a) queen, 0.5089 b) 0.6837 c) 0.5561

Table 6-5: Word similarity examples from w2v Wikipedia trained models

Both models are trained using Wikipedia datasets using different vector sizes. Although these scores are not comparable in terms of the model's accuracy, we can observe that are models learn the relationship in "a", where the answers are "queen" in both models. The similarity score of the example in "b" is higher than that of in "c" owing to the property that words that occur together are closer in semantic space, although replacing the words "woman" and "man" changes the meaning of a sentence.

6.3.6 Smaller Fragments of Text: From Characters to Vectors (ng2v)

The trigrams of the sentence *"peter pan is coming on abc family right now"* have already been shown in Table 6-2 (Section 6.3). The similarities of the first four trigrams of this sentence, "*pet ete ter er␣*" are computed using Wikipedia trained models. The similarity between the first trigram, "pet" and the other three trigrams is computed individually (Table 6-6).

Examples	Models	Similarity Scores
a) $(V_{pet,}V_{ete})$ b) $(V_{pet,}V_{ter})$ c) $(V_{pet,}V_{er\_})$	WikiModel 2	a) 0.4725 b) 0.1141 c) 0.0663
	WikiModel 3	a) 0.3941 b) 0.0287 c) 0.1096

Table 6-6: Examples from ng2v Wikipedia trained models

The similarities in Table 6-6 show that the trigram "pet" is most similar to the trigram "ete" for the both models, and the similarity score decreases with the increased vector size in WikiModel 3.

6.4 Sentence Representation with Compositions of Vectors

The construction of vectors of phrases and sentences directly from a model is still an active research area, and there have been various suggestions for sentence representations (See Chapter 2). Word vectors are extended to phrase vectors (Socher et al., 2011) and recent research shows efforts to obtain sentence, and document vectors from raw data (Le & Mikolov, 2014). However, the popularity of word vectors for sentence representation remains high due to the large number of operations can be applied between two word vectors. This is because representing sentences is task dependent. In this section, our focus is the usage of vectors to find a way of representing sentences using individual vectors of words and characters.

Vector composition can be varied for specific applications. With CBOW and SG models, it is proved that vector additions and subtractions (Mikolov, Yih, et al., 2013) perform well for similar and dissimilar words, respectively. Additionally, the two word vectors obtained from these continuous space models supports the distributional hypothesis (Firth (1957), cited from (Hirst & Mohammad, 2012)) that words that occur in similar contexts have similar meanings (Mikolov, Yih, et al., 2013). We explore such vector operations in different stages of computation. Regarding paraphrase pairs, our focus is on representing each sentence pair and individual sentences of each pair with the same set of features, in order to use them with SVM classifiers.

Recent research on the MSRPC, where distributed word embedding produces successful results (Socher et al., 2011), shows that composing word vectors retains the information to find out whether two sentences are paraphrases or not. Another approach constructs a range of sentence representations with word vectors that are drawn from semantic space models (Blacoe & Lapata, 2012) and proves that pairwise word-vector operations work well for paraphrase identification.

A recent approach to the Twitter paraphrase identification task focuses on distributed word embeddings that utilise word2vec and word2phrase algorithms (Zarrella et al., 2015) in addition to string metrics, latent semantic methods, RNN etc. These different systems are evaluated with a logistic regression algorithm. This system was ranked second in the paraphrase identification tasks of SemEval-2015 Task1. The sophistication of the

approach compared to our method, which ranked first in the same task, suggests that advanced methods do not necessarily outperform methods that are knowledge-lean.

6.4.1 Vector Operations

The core experimental unit of a paraphrase identification task is a pair of sentences, where a target sentence, S_1 that consists of n tokens, $(w_1, w_2, \dots, w_n)$, is paired up with another sentence, S_2 that consists of m tokens, $(w_1, w_2, \dots, w_m)$. The quantity of tokens (m) and (n) do not necessarily need to be equal.

Here we define a generic notation to represent sentences as sequences of vectors (Figure 6-3). This notation will be used for either word or character vectors that are derived from the statistical models. The vector of the *n'th* token of S_1 and the *m'th* token of S_2 are represented by $v_{s1_{wn}}$, $v_{s2_{wm}}$ for word vectors and $v_{s1_{ng\_n}}$,$v_{s2_{ng\_m}}$ for character vectors. Combining the features of each sentence in a pair instead of looking at them individually will make one experimental unit; the vector of a pair is $v_{p_{wk}}$, where a *P* consists of k tokens, $P_{w1}, P, \dots, P_{wk}$.

$S_1 = S_{1_{w1}}, S_{1_{w2}}, \dots, S_{1_{wn}} \rightarrow v_{s1} = v_{s1_{w1}}, v_{s1_{w2}}, \dots, v_{s1_{wn}}$
$S_2 = S_{2_{w1}}, S_{2_{w2}}, \dots, S_{2_{wm}} \rightarrow v_{s2} = v_{s2_{w1}}, v_{s2_{w2}}, \dots, v_{s2_{wm}}$
$P = P_{w1}, P_{w2}, \dots, P_{wk} \rightarrow v_p = v_{p_{w1}}, v_{p_{w2}}, \dots, v_{p_{wk}}$
$S_1 = S_{1_{ng1}}, S_{1_{ng2}}, \dots, S_{1_{ng\_n}} \rightarrow v_{s1} = v_{s1_{ng\_1}}, v_{s1_{ng\_2}}, \dots, v_{s1_{ng\_n}}$
$S_2 = S_{2_{ng1}}, S_{2_{ng2}}, \dots, S_{2_{ng\_m}} \rightarrow v_{s2} = v_{s2_{ng\_1}}, v_{s2_{ng\_2}}, \dots, v_{s2_{ng\_m}}$
$P = P_{ng1}, P_{ng2}, \dots, P_{ng\_k} \rightarrow v_p = v_{p_{ng\_1}}, v_{p_{ng\_2}}, \dots, v_{p_{ng\_k}}$

Figure 6-3: Notation for sentence representation with word and trigram vectors

Although we show the notation for words and characters above, we only use word vectors notation in the following because the vector operations discussed are exactly the same for both word and character vectors.

6.4.2 Sentence Similarity Using Word Order

Every sentence in a pair is represented as a d-dimensional vector, where d is the size of the model vocabulary. If the models do not contain the word *wi*, this word is represented as a

null vector ($v_{wt=}0$). Hence, each sentence is represented as a vector of dimension k x d, where k is size of the vocabulary and d is the model size.

$$S_1 = v_{w1}, v_{w2}, v_{wt=}0 \dots, v_{wd}$$

$$S_2 = v_{w1}, v_{w2}, v_{wt=}0 \dots, v_{wk}$$

We experimented with three different element-wise operations between word vectors: addition, multiplication and absolute value of subtraction. Representations of S_1 and S_2 with vector multiplication are denoted as $V_{mult_{s1}}$ and $V_{mult_{s2}}$.

$$V_{s1} = [(v_{w1} * v_{w2}), ((v_{w2} * v_{w3}), \dots, (v_{w(n-1)} * v_{wn})]$$

$$V_{s2} = [(v_{w1} * v_{w2}), (v_{w1} * v_{w2}), \dots, (v_{w(n-1)} * v_{wn})]$$

Sentences are represented with every two adjacent vectors derived from the model used. These operations applied between every two adjacent vectors. In this way, we simulate a type of phrasal similarity, keeping every adjacent two vectors in an order. The size of vectors is specified with the model. We then applied element-wise addition for every adjacent two vectors.

$$V_{mult_{s1}} = \sum_{i=1}^{n-1}((v_{w(i)} * v_{w(i+1)}) \quad \text{and} \quad V_{mult_{s2}} = \sum_{i=1}^{n-1}((v_{w(i)} * v_{w(i+1)})$$

The result of this operation will have the size of the vector of the model, which is defined in advance of building the models.

This method considers each sentence in a pair individually, so we will represent each sentence with a vector constructed from the result of those operations. Finally, we computed the cosine similarity of the obtained vectors.

$$Cosine1 = (V_{mult_{s1}}, V_{mult_{s2}})$$

The same method applies to addition and subtraction operations between vectors that are derived from the specified model. Although element-wise addition is symmetric and keeps the same value whether for the value of $v_{w\,(i)} + v_{w\,(i+1)}$or $v_{w\,(i+1)} + v_{w\,(i)}$, we can obtain different values with subtraction of the two inner vectors, $v_{w\,(i)} - v_{w\,(i+1)}$ which

we avoid by computing the absolute difference: $|v_{w\,(i)} - v_{w\,(i+1)}|$. Sentences S_1, S_2, are represented as $V_{add_{s1}}, V_{add_{s2}}$ for the element-wise addition of the every two inner vectors, whereas element-wise subtraction of the sentences is computed with two different variants: Sentences are denoted as $V_{sub_{s1}}, V_{sub_{s2}}$ for the subtraction of every two adjacent vectors, $v_{w\,(i)} - v_{w\,(i+1)}$, and $V_{sub2_{s1}}, V_{sub2_{s2}}$ for the absolute value of subtraction of every two adjacent vectors, $|v_{w\,(i)} - v_{w\,(i+1)}|$. The representations of each sentence in a pair are shown in Figure 6-4.

$\boldsymbol{V_{add_{s1}}} = \sum_{i=1}^{n}((v_{w(i)} + v_{w(i+1)}); \boldsymbol{V_{add_{s2}}} = \sum_{i=1}^{n}((v_{w(i)} + v_{w\,(i+1)})$
$\boldsymbol{V_{sub1_{s1}}} = \sum_{i=1}^{n}\big((v_{w(i)} - v_{w(i+1)}\big); \boldsymbol{V_{sub_{s2}}} = \sum_{i=1}^{n}\big((v_{w(i)} - v_{w(i+1)}\big)$
$\boldsymbol{V_{sub2_{s1}}} = \sum_{i=1}^{n}\big((|v_{w(i)} - v_{w(i+1)}|\big); \boldsymbol{V_{sub2_{s2}}} = \sum_{i=1}^{n}\big((|v_{w(i)} - v_{w(i+1)}|\big)$

Figure 6-4: Obtained vectors of individual sentences

Finally, we computed the cosine similarity of four different vectors of sentences, S_1 and S_2. The cosine similarity results are numbered for each operation and each is used as an input feature to the classifier Figure 6-5.

$$Cosine1 = (V_{mult_{s1}}, V_{mult_{s2}})$$
$$Cosine2 = (V_{add_{s1}}, V_{add_{s2}})$$
$$Cosine3 = (V_{sub1_{s1}}, V_{sub1_{s2}})$$
$$Cosine4 = (V_{sub2_{s1}}, V_{sub2_{s2}})$$

Figure 6-5: Cosine measures with sentence vectors

6.5 SVM Classifiers

SVMs classifiers require that each sample pair has a set of attributes, which stand for different properties, where the whole set of properties is represented as one vector. Our experiments are completed adopting 2 different classifiers: Linear and RBF kernels. We hypothesise that the ability of SVMs with RBF kernels that separate features non-linearly may be better than linear classifiers for PI tasks.

SVM classifiers are the ones provided by Scikit-learn[20] (Pedregosa et al., 2001) and both SVMs kernels have been used with their default parameters.

6.5.1 w2v and ng2v features

In section 6.4.2, cosine similarity is used for measuring the similarity of sentences in a pair. The sentence similarities are computed for each w2v and ng2v model. This results in four different similarity scores using the cosine measure. These results are used as features of an SVM classifier. We denoted each feature and usage of all features with an abbreviation, shown in Figure 6-6.

Cosine1	*cos1*
Cosine2	*cos2*
Cosine3	*cos3*
Cosine4	*cos4*
All features	*cos1234*

Figure 6-6: Features obtained from cosine similarity measure of vectors, derived from w2v and ng2v models

The features used in Chapter 4 (U, N, L1 and L2) have been combined with the features that are obtained from vector operations (Figure 6-6).

Note that these features are calculated based on the model that is being experimented on. For instance, an ng2v model is experimented on a dataset whose sentences are split into trigrams. Hence in this case the feature "N" will have the size of common character trigrams of a sentence pair, whereas "N" will have the size of common words of a sentence pair when it is computed with a w2v model.

Feature selection is a crucial part of the SVM classification process. A large quantity of features makes the feature selection process quite difficult. Our focus is to use more informative features that will help to identify paraphrase pairs, although they are fewer in quantity. Feature ablation might help in identifying the more informative features, as well as projecting their inter-correlation in a variety of combinations. As seen in Chapter

[20] http://scikit-learn.org/stable/#

4, the set of four features (U, N, L1, L2) perform well together, but the features (U, N) might give better results in some cases by removing the features L1 and L2.

6.6 Results

Our main experimental dataset is the TPC. We first experimented on the development set and then evaluated the test set of the TPC for each statistical model. The best performing statistical models from the TPC are then used in experiments with the MSRPC. The results from both datasets are analysed, individually and in comparison. Later, we compare our best results with the state-of-the-art results from both datasets, the TPC and MSRPC. For simplicity, each model's name is denoted with the attributes of the constructed model (Figure 6-7).

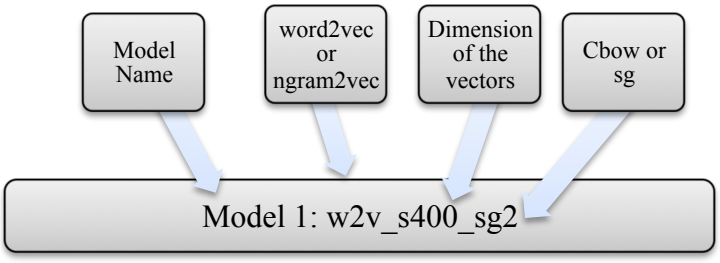

Figure 6-7: The notations for statistical models

6.6.1 TPC Training Set Model Results

In this section, we show the results obtained from the training set models. The results (Table 6-7) are obtained from the RBF Kernel and Linear Kernel, all features (cos1234 and U, N, L1, L2).

Model 1 (M1): w2v_s400_sg	**Development Set**			
	SVM	Acc.	Pre.	Rec.	F-sc.
	SVC(linear)	75.91	71.72	52.90	60.89
	SVC(rbf)	75.98	72.63	51.74	60.43
	Test Set			
	SVC(linear)	85.56	65.00	66.86	65.92
	SVC(rbf)	86.28	68.52	63.43	65.88
Model 2 (M2): ng2v_s400_sg	**Development Set**			
	SVM	Acc.	Pre.	Rec.	F-sc.
	SVC(linear)	75.33	72.66	48.74	58.34
	SVC(rbf)	74.87	72.12	47.44	57.24
	Test Set			
	SVC(linear)	85.20	64.09	66.29	65.17
	SVC(rbf)	85.44	65.14	65.14	65.14
Model 3 (M3): ng2v_s400_cbow	**Development Set**			
	SVM	Acc.	Pre.	Rec.	F-sc.
	SVC(linear)	75.50	73.04	48.94	58.61
	SVC(rbf)	74.66	71.73	47.03	56.81
	Test Set			
	SVC(linear)	84.96	63.54	65.71	64.61
	SVC(rbf)	85.80	66.09	65.71	65.90
Model 4 (M4): w2v_s400_cbow	**Development Set**			
	SVM	Acc.	Pre.	Rec.	F-sc.
	SVC(linear)	74.70	71.83	47.10	56.90
	SVC(rbf)	74.24	73.67	42.54	53.93
	Test Set			
	SVC(linear)	85.68	65.54	66.29	65.91
	SVC(rbf)	86.04	67.68	63.43	65.49

Table 6-7: TPC training set model results

The above results, in Table 6-7, are varied depending on different attributes. The development set results are good indicator of the performance of the applied methods. We highlight a few results:

- The highest result on the test set, considering both accuracy (86.28) and F-score (65.88), is obtained with Model 1 using the RBF kernel, which is indicated from the development set of M1.
- Comparing M2 and M3, the ng2v model constructed with CBOW seems to perform better than SG algorithm.
- Overall results show that the linear kernel performs better than the RBF kernel in each experiment. In addition, test set results are not distinctively different, but the

development set results from M1 are considerably higher than the other development set results.

- Comparing M1 and M2, the only difference is the training algorithm, so the SG (M1) performs better than CBOW (M2).

6.6.2 TPC Wikipedia Model Results

We first present the results from each model from both the development and test sets. We then perform feature ablation on the models which gives the highest results. The individual cosine features are then examined in order to identify their individual performance.

The results from the Wikipedia models (Table 6-8) are obtained using all features (cos1234 and UNL1L2). These models are constructed using the CBOW algorithm only. The notable results as shown:

- The highest results from the development set are obtained from WikiM2 and WikiM3, in which both models are constructed from character trigrams. The only difference between these two models is the size of vectors. The highest results obtained from WikiM2 was with RBF kernels, but the development set shows the linear kernel performs better on the test set, so it is safe to say that the larger size of vectors increases the model performance.
- The results of w2v models, WikiM1 and WikiM4, show a notable decrease compared to the results of ng2v models on the test sets, although their development set results are fairly comparable to the ng2v models.
- Overall, the performance of the RBF kernel is better than the linear kernel.

As for a general comparison of the results from Table 6-7 and Table 6-8, w2v models built from the training sets of corpora seem to outperform the w2v models built from Wikipedia dataset. However, the performance of the ng2v models trained on Wikipedia, particularly WikiM2, is higher than that of the models in both tables.

WikiM1: w2v_s200_ws5_cbow	**Development Set**			
	SVM	Acc.	Pre.	Rec.	F-sc.
	SVC(linear)	74.24	70.07	47.72	56.77
	SVC(rbf)	74.37	71.73	45.67	55.81
	Test Set			
	SVM	Acc.	Pre.	Rec.	F-sc.
	SVC(linear)	84.49	61.54	68.57	64.86
	SVC(rbf)	84.61	63.22	62.86	63.04
WikiM2: ng2v_s200_ws5_cbow	**Development Set**			
	SVM	Acc.	Pre.	Rec.	F-sc.
	SVC(linear)	75.50	73.47	48.33	58.31
	SVC(rbf)	75.28	72.98	48.06	57.95
	Test Set			
	SVM	Acc.	Pre.	Rec.	F-sc.
	SVC(linear)	85.56	65.00	66.86	65.92
	SVC(rbf)	86.52	67.42	68.57	67.99
WikiM3: ng2v_s400_ws5_cbow	**Development Set**			
	SVM	Acc.	Pre.	Rec.	F-sc.
	SVC(linear)	75.16	72.65	47.99	57.80
	SVC(rbf)	75.31	73.40	47.58	57.73
	Test Set			
	SVM	Acc.	Pre.	Rec.	F-sc.
	SVC(linear)	85.44	64.64	66.86	65.73
	SVC(rbf)	86.40	67.43	67.43	67.43
WikiM4: w2v_s400_ws5_cbow	**Development Set**			
	SVM	Acc.	Pre.	Rec.	F-sc.
	SVC(linear)	74.29	70.58	47.10	56.5
	SVC(rbf)	74.03	70.94	45.26	55.26
	Test Set			
	SVM	Acc.	Pre.	Rec.	F-sc.
	SVC(linear)	83.89	60.53	65.71	63.01
	SVC(rbf)	84.01	61.45	62.86	62.15

Table 6-8: TPC Wikipedia models results

Based on the test results of each model, we perform feature ablation on some of the models where the highest results are obtained. The highest results are obtained from the WikiM2 and WikiM3 models using all features.

Table 6-9 demonstrates the best feature combinations that lead to the best results using the WikiM2 and WikiM3 models on the TPC test set. The best result is obtained from the WikiM2 model with the four features: cos1, cos2, cos3 and N. One distinct point is the applied classifier; RBF kernels are better at classifying the paraphrase pairs.

Test Set (feature ablation)						
WikiM2	**Features**	**SVM**	**Acc.**	**Pre.**	**Rec.**	**F-sc.**
	cos123 and N	SVC(rbf)	86.87	68.36	69.14	68.75
	cos23 and UN	SVC(rbf)	86.16	65.95	69.71	67.78
	cos23 and NL1L2	SVC(rbf)	86.52	67.61	68.00	67.81
Test Set (feature ablation)						
WikiM3	**Features**	**SVM**	**Acc.**	**Pre.**	**Rec.**	**F-sc.**
	cos234 and NL1L2	SVC(rbf)	86.75	68.18	68.57	68.38
	cos123 and UN	SVC(rbf)	86.28	66.67	68.57	67.61

Table 6-9: WikiModel 2 and WikiModel 3 results of TPC after feature ablation

In Table 6-10, a comparison of individual features (cos1, cos2, cos3 and, cos4) is provided using the WikiM2 and WikiM3 models. For convenience, each cosine feature is notated along with the sign that indicates the applied vector operations. For instance, cos1 is presented as *cos1_(*)*, which is calculated by multiplying the character vectors. The result of the feature cos3 is below the baseline in each model, which suggests that the subtraction operation is not performing well when it is used individually. However, the other three features mostly perform well with both kernels. The best-performing feature is cos2 with the RBF kernel, which is obtained by adding character vectors from the WikiM3.

	Features	SVM	Acc.	Pre.	Rec.	F-sc.
WikiM2	cos1_(*)	SVC(linear)	79.55	51.72	60.00	55.56
		SVC(rbf)	81.38	55.43	55.43	55.43
	cos2_(+)	SVC(linear)	82.70	57.73	64.00	60.70
		SVC(rbf)	83.29	59.67	61.71	60.67
	cos4_(abs(-))	SVC(linear)	82.70	57.98	62.29	60.06
		SVC(rbf)	83.89	62.35	57.71	59.94
	Features	**SVM**	**Acc.**	**Pre.**	**Rec.**	**F-sc.**
WikiM3	cos1_(*)	SVC(linear)	80.67	53.40	58.29	55.74
		SVC(rbf)	81.98	56.82	57.14	56.98
	cos2_(+)	SVC(linear)	84.01	60.85	65.71	63.19
		SVC(rbf)	84.25	61.62	65.14	63.33
	cos4_(abs(-))	SVC(linear)	81.50	54.95	63.43	58.89
		SVC(rbf)	83.65	61.05	60.00	60.52

Table 6-10: Performance RBF and Linear classifiers using individual cosine features on the test set

6.6.3 Comparison of Overall Results

6.6.3.1 TPC

Our best performing results (Table 6-11) and the state-of-the-art results from the Twitter Paraphrase Corpus (Table 6-12) are shown below.

Model	Acc.	Pre.	Rec.	F-sc.
WikiM2 _SVC (RBF_{cos123 and N})	86.87	68.36	69.14	68.75
WikiM3_SVC (RBF_{cos234 and NL1L2})	86.75	68.18	68.57	68.38
$C2_{U,L1,L2}W1_N$ (RBF kernel)	86.6	67.8	68.6	68.2

Table 6-11: Our highest results on TPC

Model	Acc.	Pre.	Rec.	F-sc.
SVC (Linear kernel)_ASOBEK	86.5	68.0	66.9	67.4
(Zarrella et al., 2015)	--	56.9	80.6	66.7
(J. Zhao & Lan, 2015)	--	58.3	76.7	66.2
Baseline	--	67.9	52.0	58.9

Table 6-12: State-of-the-art results on TPC

Our best results reach F-score and accuracy values of 68.75 and 86.87 respectively. This results are obtained with the features WikiM2 _{cos123 and N} using an RBF kernel. The second highest results are obtained from WikiM3 features_{cos234 and NL1L2} with

an RBF kernel; these results are slightly lower than the best model. This reveals that the combined features of ng2v models efficiently classify paraphrase pairs.

6.6.3.2 MSRPC

The methods applied to the TPC show that the w2v and ng2v models perform well. The obtained results are evaluated by simply adjusting the same methods for the MSRPC. Here we report only the results from the ng2v models from Wikipedia, where the highest results are obtained.

Table 6-13 presents the performance of WikiM2 and WikiM3 models using all features (cos1234 and UNL1L2). The results are quite low considering the performance of previous methods (Chapter 4) that use only the features U, N, L1, L2 individually. The results are quite varied, so comparison does not seem plausible.

	Test Set (WikiModel2: ng2v_s200_ws5)					
WikiM2	**Features**	**SVM**	**Acc.**	**Pre.**	**Rec.**	**F-sc.**
	cos1234 and UNL1L2	SVC(linear)	72.64	73.82	91.19	81.59
		SVC(rbf)	73.10	74.45	90.67	81.76
	cos3 and UN	SVC(rbf)	73.62	73.99	93.03	82.43
	Test Set (WikiModel3: ng2v_s400_ws5)					
WikiM3	**Features**	**SVM**	**Acc.**	**Pre.**	**Rec.**	**F-sc.**
	cos1234 and UNL1L2	SVC(linear)	72.93	74.39	90.41	81.62
		SVC(rbf)	72.81	74.18	90.67	81.60
	cos3 and UN	SVC(rbf)	73.51	73.89	93.03	82.36
	cos34 and UNL1L2	SVC(rbf)	73.51	74.16	92.33	82.25

Table 6-13: WikiM2 and WikiM3 results from MSRPC

Table 6-14 presents the state-of-the-art-results results obtained from Neural Network approaches on the MSRPC. Our results are (WikiM2_SVC (RBF)_{cos3 and UN}:73.6 accuracy and 82.4 F-score). This is low compared to the more sophisticated methods employed using w2v algorithm, but they are still competitive and outperform Hu et al.'s (2014) sentence matching methods.

The results on the MSRPC are lower as compared to the TPC results. One of the reasons might be the difference between the two corpora in terms of the length of sentences: the sentences that are longer and constitute complex clauses in the MSRPC may

make it hard to detect paraphrases. Another reason might be that increasing the amount of training data and/or using un-annotated data may help but it may not always improve the results as much as we expect.

Model	Acc.	F-sc.
(Hu et al., 2014) (ARC-II)	69.9	80.9
(Socher et al., 2011)	76.8	83.6
(Yin & Schütze, 2015) (BI-CNN-MI)[21]	78.1	84.4
(He et al., 2015)	78.6	84.7

Table 6-14: State-of-the-art results from Neural Network Models on MSRPC

6.7 Discussion

Recent NLP research shows that the concept of continuous representations of words as vectors brings a new direction for solving most NLP problems, as well as the paraphrase identification problems. However, defining the most appropriate approach for a specified problem requires a deep exploration of a wide range of possible variations. We point out a few possibilities that might help improve our methods:

- We ignored the fact that including words that occur rarely in vocabulary might decrease the accuracy of results. Pruning words that occur less in the dataset used to construct models might help acquire more accurate representations of vectors.
- Using representations of sentence vectors with composition of word and character vectors might be a viable approach for further development of the applied methods.
- We have chosen to construct models for character trigrams only. Using different sizes of character n-grams is still worth exploring.
- Although the size of training sets of the corpora are not enough to construct accurate models, we have shown that usage of a small dataset might be utilised for languages where data sparseness is a problem. Although Wikipedia models perform well in terms of the accuracy and F-score, increasing the amount of training data by utilizing the Wikipedia dataset do not do so well as compared to the models that

[21] Using MT metrics with BI-CNN-MI, their results increased to 78.4 and 84.6 for accuracy and F-score

use only training set of TPC. Therefore, we may not know the amount of required training data while using knowledge-lean techniques.

- For the Twitter Paraphrase Corpus in particular, the development set results seem slightly better using the models constructed from the training set, in comparison to the results of our previously applied methods (Chapter 4). However, the test set results do not show an improvement in the same way as the development set results. This is because the words in the train set are mostly seen in the development set, whereas there are more unseen words in the test set, represented as null vectors.
- RBF kernel classifiers seem to perform better than linear classifiers in many experiments. This is because we use a small number of features, which leads to better performance with RBF kernels, whereas linear kernels are known to work well with a large number of features (Hsu et al., 2008).

6.8 Overall Summary

Reviewed literature shows that a variety of Neural Network (NN) approaches have been successfully applied to the paraphrase identification problem. Our work focused on building models with w2v methods, using CBOW and SG algorithms, in order to learn the vector representations of words and character trigrams. Representation of vectors are learned from a very large dataset, as well as from the training sets of corpora experimented with. These datasets are not normalised using any semantic or syntactic analysis tools; lowercasing and removal of punctuation are the only techniques performed. Punctuation is removed from Wikipedia and training sets for adaptation to the TPC.

Each constructed model, whether character-based or word-based, brings a very wide range of possibilities to be experimented with. We have tried to limit the attributes of each model in a way that can be used for comparison, and in consideration of the best possible attributes from previous experiments.

We proposed an approach for finding semantic relations of sentences from vectors of words as well as characters. We applied a variety of vector operations between vectors of words and characters in order to represent each sentence of a pair as one vector. The vector

similarities were then computed using the cosine measure, wherein the score of cosine similarity was used as an input to SVM classifiers. The features obtained from ng2v models in combination with the set of four features (U, N, L1, L2) outperform the previously applied methods.

The PI challenge on noisy user-generated text is experimented with to explore the suitability of NN models using the TPC in addition to the benchmark paraphrase corpus, the MSPRC. Despite the fact that the Wikipedia training data was used to build our models, and this differs from the TPC linguistically, we have shown that a knowledge-lean method of NN for colloquial language, such as Twitter, outperforms the previously applied methods. The MSRPC was then used to experiment with the methods where we had obtained the highest results on TPC. Although the methods did not improve over previous results on the MSRPC, it has been shown that results are competitive.

Chapter 7

7 Conclusions and Future Directions

This thesis has discussed the problem of paraphrase identification, focusing on knowledge-lean techniques. We have considered how far knowledge-lean techniques can be utilised for paraphrase identification tasks with respect to recent research. We have explored a variety of knowledge-lean methods elaborating their applicability and usefulness for identification of paraphrases.

The remainder of this chapter starts with a summary of our main results. Next, the research questions are viewed in the light of our findings. We highlight the contributions of this thesis to the research area. Finally, we conclude, drawing attention to possible future work and future directions.

7.1 Summary

Paraphrase Identification is a highly integrated field, borrowing ideas from other NLP tasks and having ideas built upon it. Paraphrasing tasks have relevance to NLP applications owing to the nature of paraphrasing, which can capture linguistic variations. A rich and extensive literature review of paraphrasing applications was provided. In particular, a comprehensive amount of research on paraphrase identification tasks was explained in relation to the knowledge-lean methods.

A gradual development of knowledge-lean techniques – from simple overlapping features to semantic features – has been shown for the paraphrase identification task. In the last table, Table 7-1, we present the best results from each model experimented with in each chapter of this thesis. We conducted three different main experiments on the Microsoft Research Paraphrase Corpus. The highest results are obtained from the RBF Kernel using character bigrams and word unigrams (C2W1) comprising a set of four features (U, N, L1

and L2). It seems that the distributional methods don't help for MSRPC but do for TPC. As shown in the table, the results from the TPC with the WikiM2 model are notably higher compared to the Linear Kernel results using the combined features C2W1. Evaluation on PAN and the TuPC showed that the set of four features with SVM classifiers out-performed the simple overlap measures. Although it is hard to draw a line between overall methods that are the result of a variety of composed parameters such as features, classifiers and models, it can be stated that the character bigram features (C2) notably work well with SVM classifiers; C2 features, which are applied to all paraphrase corpora, increase the results remarkably whether they are combined with word unigram features (MSRPC, PAN, TPC) or used alone (TuPC), and whether they are used with the RBF kernel (MSRPC, PAN) or the Linear Kernel (TPC, TuPC).

Data	Chapters	Features and Classifiers	Acc.	F-sc.
MSRPC	Chapter 3	Jaccard_Gen[22] (MSR_Tok)	74.1	82.3
	Chapter 4	**C2**W1 (RBF Kernel)	74.2	82.7
	Chapter 6	WikiM2_SVC (RBF)_{cos3 and UN}	73.6	82.4
		Baseline	66.5	79.9
PAN	Chapter 3	PAN_PoS_{all}-Jaccard	89.8	89.6
	Chapter 4	**C2**W1 (RBF Kernel)	92.4	92.3
		Baseline	88.6	87.8
TPC	Chapter 4	**C2**W1 (Linear kernel)	86.5	67.4
	Chapter 6	WikiM2 _SVC (RBF)_{cos123 and N}	86.9	68.8
		Baseline	--	58.9
TuPC	Chapter 5	Cosine (word_level)	76.2	82.2
	Chapter 5	**C2** (Linear Kernel)	77.5	83.7
		Baseline	66.4	79.8

Table 7-1: Overall results and applied methods from four different experimental datasets

7.2 Answering the Questions

How effective is the use of simple overlap methods in order to identify paraphrase pairs?

We began with the exploration of overlap features on different variants of datasets obtained from pre-processing techniques. A simple classifier based on a chosen threshold presented satisfactory results with both similarity measures and distributional similarity measures. Although the results were varied regarding the different variants of pre-processed datasets,

[22] Character bigrams features are used.

it was shown that all results obtained were over the baseline, and simple overlap measures can be used effectively for paraphrase identification problems.

Moreover, we have shown that pre-processing might not be essential for the paraphrasing task. The usage of a more advanced classifier on a dataset, where lowercasing the capital letters is the only applied pre-processing technique, in combination with more stable overlap features has outperformed the most sophisticated methods. We discussed the information loss that we witness with pre-processing.

Can knowledge-lean techniques be adapted to another language for paraphrase identification?

A quasi-answer to this question might be given by noting that knowledge-lean methods have been successfully applied to a Twitter dataset, which presents challenges in terms of the way the language is used. It contains highly colloquial and non-literary text, where there are a variety of words that are not present or defined in a dictionary. We infer that another language that is similar to English language for the computational approach such as alphabet, and space separated words, yet different with regard to syntax, might benefit from knowledge-lean methods.

This question turned out to be a particularly challenging one in terms finding a paraphrase corpus in another language for the paraphrase identification task. The limitations on the existing paraphrase corpora have been highlighted regarding their construction and annotation process. Paraphrase generation methods are combined in considering these limitations, therefore, and a paraphrase corpus of Turkish is constructed to be used with both tasks; paraphrase identification and semantic textual similarity. This corpus is then experimented with using the knowledge-lean methods that were previously conducted on the English paraphrase corpora. It was proven that Turkish paraphrases can be identified without the use of language specific tools or resources. Hence, the answer to this question raises another question about extending these methods across different languages.

To what extent can knowledge-lean techniques be used for paraphrase identification, beyond relying on overlapping features? Can semantic relations of sentences of a

paraphrase pair be defined without means of semantic tools and can this help to identify paraphrase pairs?

Continuous representation of words as vectors based on Neural Network Models is becoming highly popular in the NLP community. This underlies the concepts of DSMs models to which we draw attention in the reviewed literature. We presented a variety of knowledge-lean w2v models. Furthermore, a character level approach (ng2v) based on w2v algorithms was introduced, carrying the same attributes as w2v models. We successfully built these models using a very large dataset downloaded from the web. In addition, we utilise the training sets of corpora to find out whether a small dataset is sufficient for building models. These datasets are not normalised prior to the training process, apart from lowercasing and punctuation removal. We obtained a limited set of features by computing the similarity of sentence vectors, which are composed from vectors of words and characters. These features have proven to be informative for identifying paraphrases. Moreover, their combinations with simple overlap measures increase the performance of SVM classifiers, reaching a competitive result on the TPC using ng2v features, and showing competitive results on the MSRPC compared to many sophisticated methods.

7.3 Contributions of the Thesis

This work is innovative in terms of improving the automated understanding of a text without usage of any external resource. Moreover, our method may bring a new approach for other NLP applications that benefit from paraphrasing methods.

Being able to produce better results on two different paraphrase corpora indicates that these methods may be applied across languages. Therefore, many languages that currently lack resources might benefit from knowledge-lean techniques.

Data pre-processing might to some degree not be useful in terms of paraphrase identification because information could potentially be lost. We showed that methods with less processing – tokenisation only – on the Twitter Paraphrase Corpus can achieve high results. Furthermore, we discovered that a combined set of four features (U, N, L1, L2) derived from set theory provides a representation of a sentence pair that is useful for PI. SVM classifiers using this set of four features have proven to work well for identifying

paraphrase pairs. These features have been considered in character and word level features. The combination of these features for the task of identifying Twitter paraphrase pairs performed the best among many sophisticated methods in SemEval-2015 Task 1: Paraphrase Identification and Semantic Similarity in Twitter. Furthermore, usage of these features has proven that competitive results can be obtained with other paraphrase data: the MSRPC and PAN.

A first experimental paraphrase corpus in Turkish was constructed (TuPC). Moreover, the TuPC is designed for the sentential semantic similarity task, with a fine-grained scoring scheme made available to the NLP community. We have shown that a method requiring fewer resources might be applicable across a variety of texts or languages: this includes a colloquial dataset, Twitter, and another language, Turkish. There is still room for elaborating these methods and it would be worthwhile to extend them across other languages.

Paraphrase identification methods, which might or might not benefit from simple overlapping features such as words or character n-grams, might benefit from other knowledge-lean approaches as they do not require an annotated dataset and semantic resources. While other research focuses on the largest fragments of text such as phrases, sentences, paragraphs, and so on, for paraphrase identification, an innovative approach is presented here using character vectors. This is the only study that we are aware of operating on character vectors for the paraphrase identification task. And the results show that character vectors can be better at identifying paraphrase pairs than can word vectors.

7.4 Future directions

Although the language used in Wikipedia is not entirely formal, Wikipedia articles are still generally in a standard form. On the other hand, Twitter is not just informally written text of microblogs; it is almost like another language. The dataset we used for training comprised Wikipedia articles, and there are many words that do not appear in Wikipedia, even in millions of articles. One possible way of improving the applied methods is to use a Twitter dump file; this will at least increase the number of domain-specific words that are absent in the Wikipedia dataset. Another improvement might be to train the models with

extracted sentences instead of using long Wikipedia articles. This will require defining a small window size of words, which will limit the number of neighbour words, while focusing more on the words that occur more frequently. In addition to this, a sentence-level dataset to train character n-gram models might be more beneficial for capturing more reliable relations between character vectors.

The construction process of a paraphrase corpus that requires a minimum amount of common words between sentence pairs eliminates the type of pairs that have no common words, but convey the same meaning. A future development to overcome this problem might be the use of the w2v algorithm for pairing the ideal sentences by looking at highly similar vectors, instead of overlapping items. Their combination might even be more advantageous. The idea of constructing a paraphrase corpus beyond relying on lexical overlaps is an intriguing approach that could be explored in further research.

Although the available dataset from Wikipedia in Turkish is limited, it is worth exploring the applicability of w2v and ng2v statistical models to the TuPC. Also, it would be worth exploring the relationship between quantity of data and accuracy by plotting learning curves. This will particularly help when considering possible approaches for a less resourced language. Furthermore, the usage of character level vectors has been proposed in a few of the latest studies. In Ling et al.'s (2015) study, word representations are composed of characters of vectors, called character to word (C2W), for language modelling and PoS-tagging without using any handcrafted features. They experimented on five different languages, including English and Turkish. While their results are competitive for English as compared to the Stanford PoS-tagger, they achieved state-of-the-art results, producing a remarkable improvement in results from the Turkish language. Kim, Jernite, Sontag, and Rush (2015) apply a simple convolutional neural network model, which uses character level inputs for word representations. Again, this method outperforms the models that use word/morpheme level features in morphologically rich languages, besides having competitive results in English.

Knowledge-lean methods might help improve the accuracy of machine translation and text summarisation applications. Real-life applications, such as plagiarism

identification systems, may also be improved by better paraphrase identification methods and this would be worth exploring in future.

In this research, we have emphasized the use of knowledge-lean methods for identifying sentential paraphrases, a task which requires judgements of semantic equivalency between a pair of sentences. We argue that unless we explore the strengths of knowledge-lean techniques, we will not be able to explore what is really needed in terms of tools and resources. Our findings have shown that methods relying only on text-based statistics, without the usage of semantic tools or resources can be very powerful for the task of paraphrase identification.

Bibliography

Agirre, E., Banea, C., Cardie, C., Cer, D., Diab, M., Gonzalez-Agirre, A., … Wiebe, J. (2014). SemEval-2014 Task 10: Multilingual Semantic Textual Similarity. In *Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014)* (pp. 81–91). ACL.

Agirre, E., Banea, C., Cardie, C., Cer, D., Diab, M., Gonzalez-Agirre, A., … Wiebe, J. (2015). SemEval-2015 Task 2 : Semantic Textual Similarity, English, Spanish and Pilot on Interpretability. In *SemEval2015* (pp. 252–263).

Agirre, E., Cer, D., Diab, M., & Gonzalez-Agirre, A. (2012). Semeval-2012 task 6: A pilot on semantic textual similarity. In *Proceedings of the 6th International Workshop on Semantic Evaluation, in conjunction with the First Joint Conference on Lexical and Computational Semantics* (pp. 385–393).

Agirre, E., Cer, D., Diab, M., Gonzalez-Agirre, A., & Guo, W. (2013). *SEM 2013 shared task : Semantic Textual Similarity. In *The Second Joint Conference on Lexical and Computational Semantics (*SEM 2013)* (Vol. 1, pp. 32–43).

Androutsopoulos, I., & Malakasiotis, P. (2010). A Survey of Paraphrasing and Textual Entailment Methods. *Artificial Intelligence Research*, *38*(1), 135–187.

Bannard, C., & Callison-Burch, C. (2005). Paraphrasing with Bilingual Parallel Corpora. In *Proceedings of the 43th Annual Meeting on Association for Computational Linguistics* (pp. 597–604).

Baroni, M., Dinu, G., & Kruszewski, G. (2014). Don't count, predict! A systematic comparison of context-counting vs. context-predicting semantic vectors. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics* (Vol. 1, pp. 238–247). Baltimore, Maryland.

Barron-Cedeno, A., Vila, M., Marti, M. A., & Rosso, P. (2013). Plagiarism Meets Paraphrasing: Insights for the Next Generation in Automatic Plagiarism Detection.

Computational Linguistics, *39*(4), 917–947. http://doi.org/http://dx.doi.org/10.1162/COLI_a_00153

Barzilay, R., & Lee, L. (2003). Learning to Paraphrase: An Unsupervised Approach Using Multiple-Sequence Alignment. In *Naacl-2003* (pp. 16–23). http://doi.org/10.3115/1073445.1073448

Barzilay, R., & McKeown, K. R. (2001). Extracting paraphrases from a parallel corpus. In *Proceedings of the 39th Annual Meeting on Association for Computational Linguistics - ACL '01* (pp. 50–57).

Barzilay, R., Mckeown, K. R., & Elhadad, M. (1999). Information Fusion in the Context of Multi-Document Summarization. In *Proceedings of ACL* (pp. 550–557).

Bhagat, R., & Ravichandran, D. (2008). Large Scale Acquisition of Paraphrases for Learning Surface Patterns. In *Proceedings of ACL-08: HLT* (pp. 674–682).

Bird, S., Loper, E., & Klein, E. (2009). *Natural Language Processing with Python*. O'Reilly Media Inc.

Blacoe, W., & Lapata, M. (2012). A Comparison of Vector-based Representations for Semantic Composition. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL '12)* (pp. 546–556).

Briscoe, T., Carroll, J., & Watson, R. (2006). The Second Release of the RASP System. In *Proceedings of the COLING/ACL 2006 Interactive Presentation Sessions* (pp. 77–80). Sydney, Australia.

Brockett, C., & Dolan, W. B. (2005). Support Vector Machines for Paraphrase Identification and Corpus Construction. In *Third International Workshop on Paraphrasing (IWP2005)* (pp. 1–8). Jeju, Republic of Korea: Asia Federation of Natural Language Processing.

Brockett, C., & Kok, S. (2010). Hitting the Right Paraphrases in Good Time. In *Annual Conference of the North American Chapter of the Association for Computational*

Linguistics.

Callison-Burch, C. (2008). Syntactic Constraints on Paraphrases Extracted from Parallel Corpora. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing* (pp. 196–205).

Callison-Burch, C., Cohn, T., & Lapata, M. (2008). ParaMetric : An Automatic Evaluation Metric for Paraphrasing. In *Proceedings of the 22nd International Conference on Computational Linguistics* (pp. 97–104).

Callison-Burch, C., Koehn, P., & Osborne, M. (2006). Improved Statistical Machine Translation Using Paraphrases. In *Proceedings of the Human Language Technology Conference of the North American Chapter of the Association of Computational Linguistics (HLT-NAACL'06)* (pp. 17–24).

Can, F., Kocberber, S., Balcik, E., Kaynak, C., Ocalan, H. C., & Vursavas, O. M. (2008). Information retrieval on Turkish texts. *Journal of the American Society for Information Science and Technology*, *59*(3), 407–421. http://doi.org/10.1002/asi

Cavnar, W. B., & Trenkle, J. M. (1994). N-Gram-Based Text Categorization. In *Proceedings of SDAIR-94, 3rd Annual Symposium on Document Analysis and Information Retrieval* (pp. 161–175).

Chang, C., & Lin, C. (2011). LIBSVM : A Library for Support Vector Machines. *ACM Transactions on Intelligent Systems and Technology (TIST)*, *2*(3), 1–27.

Chen, D. L., & Dolan, W. B. (2011). Collecting Highly Parallel Data for Paraphrase Evaluation. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies (HLT'11)* (pp. 190–200).

Cohen, J. (1960). A coefficient of agreement for Nominal Scales. *Educational and Psyhological Measurement*, *20*(1), 37–46.

Cohn, T., Callison-Burch, C., & Lapata, M. (2008). Constructing Corpora for the Development and Evaluation of Paraphrase Systems. *Computational Linguistics*, *34*(4), 597–614. http://doi.org/http://dx.doi.org/10.1162/coli.08-003-R1-07-044

Corley, C., & Mihalcea, R. (2005). Measuring the semantic similarity of texts. In *Proceedings of the ACL Workshop on Empirical Modeling of Semantic Equivalence and Entailment (EMSEE '05)* (pp. 13–18).

Culicover, P. W. (1968). Paraphrase Generation and Information Retrieval from Stored Text. *Mechanical Translation and Computational Linguistics*, *11*(1–2), 78–88.

Das, D., & Smith, N. A. (2009). Paraphrase Identification as probabilistic quasi-synchronous recognition. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: ACL-IJCNLP '09* (Vol. 1, pp. 468–476).

Demir, S., El-Kahlout, I. D., Unal, E., Kaya, H., & El-kahlout, D. (2012). Turkish Paraphrase Corpus. In *Proceedings of the Eight International Conference on Language Resources and Evaluation (LREC'12)* (pp. 4087–4091).

Derczynski, L., Ritter, A., Clark, S., & Bontcheva, K. (2013). Twitter part-of-speech tagging for all: Overcoming sparse and noisy data. In *Proceedings of the Recent Advances in Natural Language Processing*.

Dolan, W. B., Quirk, C., & Brockett, C. (2004). Unsupervised Construction of Large Paraphrase Corpora : Exploiting Massively Parallel News Sources. In *Proceedings of the 20th International Conference on Computational Linguistics, COLING '04*. Geneva, Switzerland.

Duclaye, F., Yvon, F., Collin, O., R, F. T., Marzin, P., & Cedex, L. (2002). Using the Web as a Linguistic Resource for Learning Reformulations Automatically. In *Proceedings of the third international conference on language resources and evaluation (LREC'02)* (pp. 390–396). Las Palmas, Canary Islands, Spain.

Eyecioglu, A., & Keller, B. (2015). ASOBEK : Twitter Paraphrase Identification with Simple Overlap Features and SVMs. In *Proceedings of the 9th International Workshop on Semantic Evaluation (SemEval 2015)* (pp. 64–69). Denver, Colorado.

Eyecioglu, A., & Keller, B. (2016). Constructing A Turkish Corpus for Paraphrase

Identification and Semantic Similarity. In *Proceedings of the 17th International Conference on Intelligent Text Processing and Computational Linguistics. Lecture Notes in Computer Science* (Vol. 9623, pp. 562–574).

Fellbaum, C. (1998). *WordNet: An electronic lexical database.* Cambridge, MA: MIT Press.

Fernando, S., & Stevenson, M. (2008). A Semantic Similarity Approach to Paraphrase Detection. In *Proceedings of the 11th Annual Research Colloquium of the UK Special Interest Group for Computational Linguistics* (pp. 45–52).

Finch, A., Hwang, Y.-S., & Sumita, E. (2005). Using Machine Translation Evaluation Techniques to Determine Sentence-level Semantic Equivalence. In *Proceedings of the Third International Workshop on Paraphrasing (IWP2005)* (pp. 17–24).

Firth, J. R. (1957). A synopsis of linguistic theory 1930-55. *Studies in Linguistic Analysis (Special Volume of the Philological Society)*, 1–32.

Fleiss, J. L. (1971). Measuring Nominal Scale Agreement Among Many Raters. *Psychological Bulletin*, *76*, 378–382.

Ganitkevitch, J., Callison-Burch, C., Napoles, C., & Durme, B. Van. (2011). Learning Sentential Paraphrases from Bilingual Parallel Corpora for Text-to-Text Generation. *Computational Linguistics*, 1168–1179.

Ganitkevitch, J., Van Durme, B., & Callison-Burch, C. (2013). PPDB : The Paraphrase Database. In *Proceedings of NAACL-HLT* (pp. 758--764). Atlanta,Gerogia.

Glickman, O., & Dagan, I. (2003). Identifying lexical paraphrases from a single corpus. In *Proceedings of the International Conference RANLP-2003 (Recent Advances in Natural Language Processing)* (pp. 166–173).

Guo, W., & Diab, M. (2012). Modeling Sentences in the Latent Space. In *Proceedings of the 50th Annual Meeting of Association for Computational Linguistics* (pp. 864–872).

Gwet, K. L. (2012). *Handbook of Inter-rater reliability* (Third Edit). Gaithersburg:

Advanced Analytics.

Halevy, A., Norvig, P., & Pereira, F. (2009). The Unreasonable Effectiveness of Data. *IEEE Intelligent Systems*, *24*(2), 8–12. http://doi.org/10.1109/MIS.2009.36

Han, B., & Baldwin, T. (2011). Lexical Normalisation of Short Text Messages : Makn Sens a #twitter. In *Computational Linguistics* (pp. 368–378). Portland,Oregon. http://doi.org/10.1145/0000000.0000000

He, H., Gimpel, K., & Lin, J. (2015). Multi-Perspective Sentence Similarity Modeling with Convolutional Neural Networks. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing* (pp. 1576–1586). Lisbon, Portugal.

He, W., Zhao, S., Wang, H., & Liu, T. (2011). Enriching SMT Training Data via Paraphrasing. In *Proceedings of the 5th International Joint Conference on Natural Language Processing, IJCNLP 2011* (pp. 803–810). Asian Federation of Natural Language Processing.

Hearst, M. A., & Grefenstette, G. (1992). Refining Automatically-Discovered Lexical Relations: Combining Weak Techniques for Stronger Results. In *Statistically-Based Natural Language Programming Techniques, Papers from the 1992 AAAI Workshop* (pp. 64–72). Menlo Park, CA.

Hirst, G. (2003). Paraphrasing Paraphrased. In *Invited talk at the Second International Workshop on Paraphrasing: Paraphrase Acquisition and Applications (IWP2003)*. Sapporo.

Hirst, G., & Mohammad, S. (2012). Distributional Measures of Semantic Distance : A Survey. *CoRR*, *1*, 1–39.

Hsu, C.-W., Chang, C.-C., & Lin, C.-J. (2008). A Practical Guide to Support Vector Classification. *BJU International*, *101*(1), 1396–400.

Hu, B., Lu, Z., Li, H., & Chen, Q. (2014). Convolutional neural network architectures for matching natural language sentences. *Advances in Neural Information Processing Systems*, 2042–2050.

Hunter, J. D. (2007). Matplotlib: A 2D graphics environment. *Computing In Science and Engineering*, *9*(3), 90–95.

Ibrahim, A., Katz, B., & Lin, J. (2003). Extracting structural paraphrases from aligned monolingual corpora. In *Proceedings of the second international workshop on Paraphrasing* - (pp. 57–64). Morristown, NJ, USA: Association for Computational Linguistics. http://doi.org/10.3115/1118984.1118992

Islam, A., & Inkpen, D. (2007). Semantic Similarity of Short Texts. In *Proceedings of the International conference on Recent Advances in Natural Language Processings (RANLP)*. Bulgaria.

Ji, Y., & Eisenstein, J. (2013). Discriminative Improvements to Distributional Sentence Similarity. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing* (pp. 891–896). Seattle, Washington, USA: Association for Computational Linguistics.

Kim, Y., Jernite, Y., Sontag, D., & Rush, A. M. (2015). Character-Aware Neural Language Models. *CoRR*, *1508.06615*.

Kozareva, Z., & Montoyo, A. (2006). Paraphrase Identification on the Basis of Supervised Machine Learning Techniques. In *Proceedings of the 5th International Conference on Natural Language Processing (FinTAL 2006), Lecture Notes in Artificial Intelligence* (pp. 524–533). Turku, Finland.

Le, Q., & Mikolov, T. (2014). Distributed Representations of Sentences and Documents. *International Conference on Machine Learning - ICML 2014*, *32*, 1188–1196.

Lin, D., & Pantel, P. (2001). DIRT – Discovery of Inference Rules from Text. In *Proceedings of the ACM SIGKDD Conference on Knowledge Discovery and Data Mining* (pp. 323–328).

Ling, W., Luís, T., Marujo, L., Astudillo, R. F., Amir, S., Dyer, C., … Trancoso, I. (2015). Finding Function in Form: Compositional Character Models for Open Vocabulary Word Representation. In *Proceedings of the 2015 Conference on Empirical Methods*

in Natural Language Processing (pp. 1520–1530).

Lintean, M., & Rus, V. (2010). Paraphrase Identification Using Weighted Dependencies and Word Semantics. In *Proceedings of the 22nd International Florida Artificial Intelligence Research Society Conference.* (pp. 260–265). Sanibel Island, FL.

Lintean, M., & Rus, V. (2011). Dissimilarity Kernels for Paraphrase Identification. In *Proceedings of the 24th International Florida Artificial Intelligence Research Society Conference.* (pp. 263–268). Palm Beach, FL.

Lintean, M., Rus, V., & Graesser, A. C. (2008). Using Dependency Relations to Decide Paraphrasing. In *Proceedings of the Society for Text and Discourse Conference.* Memphis, TN.

Madnani, N., Ayan, N. F., Resnik, P., Dorr, B. J., & Park, C. (2007). Using Paraphrases for Parameter Tuning in Statistical Machine Translation. In *Proceedings of the Second Workshop on Statistical Machine Translation (WMT'07).* (Vol. 20742). Prague, Czech Republic.

Madnani, N., & Dorr, B. J. (2010). Generating Phrasal and Sentential Paraphrases: A Survey of Data-Driven Methods. *Computational Linguistics*, *36*(3), 341–387. http://doi.org/10.1162/coli_a_00002

Madnani, N., Tetreault, J., & Chodorow, M. (2012). Re-examining Machine Translation Metrics for Paraphrase Identification. In *Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT'12)* (pp. 182–190). PA, USA.

Malakasiotis, P. (2009). Paraphrase Recognition Using Machine Learning to Combine Similarity Measures. In *Proceedings of the ACL-IJCNLP 2009 Student Research Workshop* (pp. 27–35). Suntec, Singapore.

Marton, Y. (2010). Improved Statistical Machine Translation with Hybrid Phrasal Paraphrases Derived from Monolingual Text and a Shallow Lexical Resource. In *The Ninth Conference of the Association for Machine Translation in the Americas*

(AMTA). Denver, Colorado.

Marton, Y., Callison-Burch, C., & Resnik, P. (2009). Improved Statistical Machine Translation Using Monolingually-Derived Paraphrases. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Singapore.

Marton, Y., Kholy, A. El, & Habash, N. (2011). Filtering antonymous, trend-contrasting, and polarity-dissimilar distributional paraphrases for improving statistical machine translation. In *Proceedings of the Sixth Workshop on Statistical Machine Translation* (pp. 237–249). Edingburgh, Scotland: Association for Computational Linguistics.

Max, A., & Wisniewski, G. (2010). Mining Naturally-occurring Corrections and Paraphrases from Wikipedia's Revision History. *Proceedings of LREC*, 3143–3148.

Mayfield, J., & Mcnamee, P. (2003). Single N-gram Stemming. In *Proceedings of the 26th annual international ACM SIGIR conference on Research and development in informaion retrieval(SIGIR '03)* (pp. 415–416). New York,USA: ACM.

Mckeown, K. R. (1983). Paraphrasing Questions Using Given and New Information. *Computational Linguistics*, *9*(1), 1–10.

Mihalcea, R., Corley, C., & Strapparava, C. (2006). Corpus-based and Knowledge-based Measures of Text Semantic Similarity. In *Proceedings of the 21st national conference on Artificial intelligence- Volume 1* (pp. 775–780). AAAI Press.

Mikolov, T., Chen, K., Corrado, G., & Dean, J. (2013). Distributed Representations of Words and Phrases and their Compositionality. In *Advances in Neural Information Processing systems* (pp. 3111–3119).

Mikolov, T., Corrado, G., Chen, K., & Dean, J. (2013). Efficient Estimation of Word Representations in Vector Space. *Proceedings of the International Conference on Learning Representations (ICLR 2013)*.

Mikolov, T., Yih, W., & Zweig, G. (2013). Linguistic regularities in continuous space word representations. *Proceedings of NAACL-HLT*, (June), 746–751.

O'Connor, B., Krieger, M., & Ahn, D. (2010). TweetMotif : Exploratory Search and Topic Summarization for Twitter. In *Proceedings of the Fourth International AAAI Conference on Weblogs and Social Media* (pp. 384–385). Association for the Advancement of Artificial Intelligence.

Owczarzak, K., Gorves, D., Genabith, J. Van, & Way, A. (2006). Contextual Bitext-Derived Paraphrases in Automatic MT Evaluation. In *StatMT '06* (pp. 86–93). Stroudsburg, PA, USA.

Pasca, M., & Dienes, P. (2005). Aligning Needles in a Haystack : Paraphrase Acquisition Across the Web. In *Proceedings of the Second international joint conference on Natural Language Processing (IJCNLP'05)* (pp. 119–130). Berlin.

Pedersen, T., & Bruce, R. (1998). Knowledge lean word-sense disambiguation. In *Proceedings of the Fifteenth National Conference on Artificial Intelligence* (pp. 800–805). AAAI Press.

Pedersen, T., Patwardhan, S., & Michelizzi, J. (2004). WordNet :: Similarity - Measuring the Relatedness of Concepts. In *American Association for Artificial Intelligence* (pp. 1024–1025).

Pedregosa, F. and Varoquaux, G. and Gramfort, A. and Michel, V., and Thirion, B. and Grisel, O. and Blondel, M. and Prettenhofer, P., and Weiss, R. and Dubourg, V. and Vanderplas, J. and Passos, A. and, & Cournapeau, D. and Brucher, M. and Perrot, M. and Duchesnay, E. (2001). Scikit-learn: Machine Learning in Python. Retrieved from http://scikit-learn.org/stable/

Power, R., & Scott, D. (2005). Automatic generation of large-scale paraphrases. In *Proceedings of the 3rd International Workshop on Paraphrasing (IWP2005)* (pp. 33–40). Jeju, Republic of Korea.

Qiu, L., Kan, M.-Y., & Chua, T.-S. (2006). Paraphrase Recognition via Dissimilarity Significance Classification. In *Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing (EMNLP '06)* (pp. 18–26). Stroudsburg, PA, USA. http://doi.org/10.3115/1610075.1610079

Quirk, C., Brockett, C., & Dolan, W. B. (2004). Monolingual Machine Translation for Paraphrase Generation. In *EMNLP-2014* (pp. 142–149).

Ravichandran, D., & Hovy, E. (2002). Learning Surface Text Patterns for a Question Answering System. In *Proceedings of the 40th Annual Meeting of the Association for Computional Linguistics*.

Rehurek, R., & Sojka, P. (2010). Software framework for topic modelling with large corpora. *Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks*, 45–50.

Resnik, P., & Smith, N. a. (2003). The Web as a Parallel Corpus. *Computational Linguistics*, *29*(3), 349–380. http://doi.org/10.1162/089120103322711578

Ritter, A., Clark, S., Etzioni, M., & Etzioni, O. (2011). Named entity recognition in tweets: an experimental study. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP'11)* (pp. 1524–1534). Stroudsburg, PA, USA.

Rus, V., Banjade, R., & Lintean, M. (2014). On Paraphrase Identification Corpora. In *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14)*. Reykjavik, Iceland: European Language Resources Association (ELRA).

Rus, V., Lintean, M., Banjade, R., Niraula, N., & Stefanescu, D. (2013). SEMILAR : The Semantic Similarity Toolkit. *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics*, 163–168.

Rus, V., Mccarthy, P. M., Lintean, M. C., Mcnamara, D. S., & Graesser, A. C. (2008). Paraphrase Identification with Lexico-Syntactic Graph Subsumption. In D. Wilson & H. C. Lane (Eds.), *FLAIRS Conference* (pp. 201–206). AAAI Press.

Shinyama, Y., & Sekine, S. (2003). Paraphrase Acquisition for Information Extraction. In *Proceedings of the second international workshop on Paraphrasing (PARAPHRASE '03), Vol. 16* (pp. 65–71). Stroudsburg, PA, USA.

Shinyama, Y., Sekine, S., & Sudo, K. (2002). Automatic paraphrase acquisition from news

articles. *Proceedings of the Second International Conference on Human Language Technology Research*, 313–318. http://doi.org/10.3115/1289189.1289218

Socher, R., Huang, E. H., Pennington, J., Ng, A. Y., & Manning, C. D. (2011). Dynamic Pooling and Unfolding Recursive Autoencoders for Paraphrase Detection. *Advances in Neural Information Processing Systems*, 801–809.

Stanovsky, G. (2012). *A Study in Hebrew Paraphrase Identification*. Ben-Gurion University of Negev.

Suen, C. Y. (1979). n-Gram Statistics for Natural Language Understanding and Text Processing. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *PAMI-1*(2), 164–172. http://doi.org/10.1109/TPAMI.1979.4766902

Tomuro, N., & Lytinen, S. L. (2001). Selecting Features for Paraphrasing Question Sentences. In *Proceedings of the Workshop on Automatic Paraphrasing at Natural Language Processing Pacific Rim Symposium (NLPRS),* (Vol. 0, pp. 55–62).

Wan, S., Dras, M., & Dale, R. (2006). Using Dependency-Based Features to Take the 'Para-farce' out of Paraphrase. In *Proceedings of the Australasian Language Technology Workshop* (pp. 131–138). Sydney, Australia.

Wu, D. (2005). Recognizing Paraphrases and Textual Entailment using Inversion Transduction Grammars. In *ACL-2005 Workshop on Empirical Modeling of Semantic Equivalence and Entailment.* (pp. 25–30). Ann Arbor, MI.

Wubben, S., Bosch, A. Van Den, & Krahmer, E. (2010). Creating and using large monolingual parallel corpora for sentential paraphrase generation. In *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14)* (pp. 4295–4299). European Language Resources Association (ELRA).

Xu, W. (2014). *Data-driven approaches for paraphrasing across language variations*. New York University.

Xu, W., Callison-Burch, C., & Dolan, W. B. (2015). SemEval-2015 Task 1: Paraphrase and semantic similarity in Twitter (PIT). In *Proceedings of the 9th International Workshop*

on Semantic Evaluation (SemEval).

Xu, W., Ritter, A., Callison-Burch, C., Dolan, W. B., & Ji, Y. (2014). Extracting Lexically Divergent Paraphrases from Twitter. *Transactions of the Association for Computational Linguistics*, *2*, 435–448.

Yang, S., Zhu, H., Apostoli, A., & Cao, P. (2007). N-gram Statistics in English and Chinese : Similarities and Differences. In *Proceedings of First IEEE International Conference on Semantic Computing* (pp. 454–460).

Yin, W., & Schütze, H. (2015). Convolutional Neural Network for Paraphrase Identification. In *The 2015 Annual Conference of the North American Chapter of the ACL* (pp. 901–911).

Zarrella, G., Henderson, J., Merkhofer, E. M., & Strickhart, L. (2015). MITRE : Seven Systems for Semantic Similarity in Tweets. In *Proceedings of the 9th International Workshop on Semantic Evaluation (SemEval 2015)* (pp. 12–17). Denver, Colorado.

Zhang, Y., & Patrick, J. (2005). Paraphrase Identification by Text Canonicalization. In *Proceedings of the Australasian Language Technology Workshop* (pp. 160–166). Sydney, Australia.

Zhao, J., & Lan, M. (2015). ECNU : Leveraging Word Embeddings to Boost Performance for Paraphrase in Twitter. In *Proceedings of the 9th International Workshop on Semantic Evaluation (SemEval 2015)* (pp. 34–39). Denver, Colorado.

Zhao, S., Lan, X., Liu, T., & Li, S. (2009). Application-driven statistical paraphrase generation. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 2 - ACL-IJCNLP '09* (Vol. 2, pp. 834–842). Suntec, Singapore: Association for Computational Linguistics. http://doi.org/10.3115/1690219.1690263

Appendix A

Chapter 3	Measures	Data			Training Data	Features
	Dice Cosine Jaccard Jaccard_Gen Dice	Data Normalisation	MSR_Tok / PAN_Tok	Tokenized-only	Training sets of MSRPC/PAN	Character Bigram Features and Lexical Features
			MSR_Lemma /PAN_Lemma	Lemmatisation is applied on tokenised-only data		
			MSR_Stem/ PAN_Stem	Stemming is applied on tokenised-only data		
		PoS tagging	MSR_PoS/ PAN_PoS	Part-of-speech tagged data	Training sets of MSRPC/PAN	Lexical Features
			MSR_PoS_{all}/ PAN_PoS_{all}	Simplified form of MSR_PoS; includes all word categories		
			MSR_PoS_(major}/ PAN_PoS_(major}	Simplified form of MSR_PoS; includes only major categories		
	Sum1, Sum2, Sum3 Max1, Max2, Max3	PoS tagging	MSR_PoS	Part-of-speech tagged data	Byblo Thesaurus	Word-word similarities
			MSR_PoS_{all}	Simplified form of MSR_PoS; includes all word categories		
			MSR_PoS_(major}	Simplified form of MSR_PoS; includes only major categories		

Chapter 4	Measures	Data	Training Data	Features
	Dice,Cosine Jaccard,Jaccard_Gen	TPC	Training sets of TPC	Lexical Features Character bigram features
	SVM (Linear and RBF kernels)			Logical operations AND, OR, XOR
				Set theory features (U, N, L1 and L2) for word unigram/bigram (W1/W2) and character unigram/bigram (C1, C2)
				Set theory features (U, N, L1 and L2) for character trigram/fourgram (C3, C4)

Chapter 5	Measures	Data	Training Data	Features
	Dice,Cosine Jaccard,Jaccard_Gen	TuPC	Training sets of TuPC	Lexical Features Character Bigram Features
	SVC (Linear and RBF kernels)			Set theory features (U, N, L1 and L2) for word unigram/bigram (W1/W2) and character unigram/bigram (C1, C2)

Chapter 6	MODELS	Data	Training data	Features	Measures
	Model 1 (M1: w2v_s400_sg)	TPC	Training sets of TPC	Set theory features (U, N, L1, L2) and cosine similarity (cos1, cos2, cos3 and cos4)	SVM (Linear and RBF kernels)
	Model 2 (M2: ng2v_s400_sg)				
	Model 3 (M3: ng2v_s400_cbow)				
	Model 4 (M4: w2v_s400_cbow)				
	WikiModel 1 (WikiM1: w2v_s200_ws5_cbow)	TPC	Wikipedia		
	WikiModel 2 (WikiM2: ng2v_s200_ws5_cbow)	TPC and MSRPC			
	WikiModel 3 (WikiM3: ng2v_s400_ws5_cbow)				
	WikiModel 4 (WikiM4: w2v_s400_ws5_cbow)	TPC			

Table 0-1: Table of system names, data, features and measures used throughout the thesis.

Character n-grams results of dice coefficient measure:

Microsoft Paraphrase Corpus		Accuracy	Precision	Recall	F-Score
Dice Coefficient	(bigrams)	73.0	73.6	93.3	82.3
	(trigrams)	72.4	75.1	88.1	81.1
	(fourgrams)	71.6	75.9	84.6	80.6

Table 0-2: Character n-grams (up to fourgrams) results of dice coefficient measure

Increasing n-gram length tends to increase precision, but with a loss of recall and overall accuracy.

Appendix B

Our team is called “ASOBEK”. There are two runs that we submitted for the SemEval Task 1 and Chapter 4 explains the details of the first run (01_svckernel).

SemEval-2015 Task 1: Paraphrase Identification and Semantic Similarity Official Evaluation Results.

SemEval-PIT-2015-results

TeamRank		TEAM	RUN	task 1 - Paraphase Identification				task 2 - Semantic Similarity				
task 1	task 2			Rank-F1	F1	Precision	Recall	Rank-Pearson	Pearson	maxF1	mPrecision	mRecall
1		ASOBEK	01_svckernel	1	0.674	0.680	0.669	18	0.475	0.616	0.732	0.531
	8	ASOBEK	02_linearsvm	2	0.672	0.682	0.663	14	0.504	0.663	0.723	0.611
2	1	MITRE	01_ikr	3	0.667	0.569	0.806	1	0.619	0.716	0.750	0.686
3		ECNU	02_nnfeats	4	0.662	0.767	0.583					
4		FBK-HLT	01_voted	5	0.659	0.685	0.634	19	0.462	0.607	0.551	0.674
5		TKLBLIIR	02_gs0105	5	0.659	0.645	0.674					
		MITRE	02_bieber	7	0.652	0.559	0.783	2	0.612	0.724	0.753	0.697
6		HLTC-UST	02_run2	7	0.652	0.574	0.754	6	0.545	0.669	0.738	0.611
	3	HLTC-UST	01_run1	9	0.651	0.594	0.720	5	0.563	0.676	0.697	0.657
		ECNU	01_mlfeats	10	0.643	0.754	0.560					
7	4	AJ-SEVAL	01_first	11	0.622	0.523	0.766	7	0.527	0.642	0.571	0.731
8	5	DEPTH	02_modelx23	12	0.619	0.652	0.589	8	0.518	0.636	0.602	0.674
9	9	CDTDS	01_simple	13	0.613	0.547	0.697	15	0.494	0.626	0.675	0.583
		CDTDS	02_simplews	14	0.612	0.542	0.703	16	0.491	0.624	0.589	0.663
		DEPTH	01_modelh22	15	0.610	0.647	0.577	13	0.505	0.638	0.642	0.634
	10	FBK-HLT	02_multilayer	16	0.606	0.676	0.549	17	0.480	0.604	0.504	0.754
10		ROB	01_all	17	0.601	0.519	0.714	10	0.513	0.612	0.721	0.531
11		EBIQUITY	01_run	18	0.599	0.651	0.554					
		TKLBLIIR	01_gsc054	19	0.590	0.461	0.817					
		EBIQUITY	02_run	19	0.590	0.646	0.543					
		BASELINE	**logistic reg**	**21**	**0.589**	**0.679**	**0.520**	**11**	**0.511**	**0.601**	**0.674**	**0.543**
12	11	columbia	02_ormf	22	0.588	0.593	0.583	20	0.425	0.599	0.623	0.577
13	12	HASSY	01_train	23	0.571	0.449	0.783	22	0.405	0.645	0.657	0.634
14		RTM-DCU	01_PLSSVR	24	0.562	0.859	0.417	4	0.564	0.678	0.649	0.709
		columbia	01_ormf	25	0.561	0.831	0.423	20	0.425	0.599	0.623	0.577
		HASSY	02_traindev	25	0.551	0.423	0.789	22	0.405	0.629	0.648	0.611
	2	RTM-DCU	02_SVR	27	0.540	0.883	0.389	3	0.570	0.693	0.695	0.691
		BASELINE	**WTMF**	**28**	**0.536**	**0.450**	**0.663**	**26**	**0.350**	**0.587**	**0.570**	**0.606**
	6	ROB	02_all	29	0.532	0.388	0.846	9	0.515	0.616	0.685	0.560
	7	MATHLING	02_twimash	30	0.515	0.364	0.880	11	0.511	0.650	0.648	0.651
15		MATHLING	01_twiemb	30	0.515	0.454	0.594	27	0.229	0.562	0.638	0.503
16		YAMRAJ	01_google	32	0.496	0.725	0.377	25	0.360	0.542	0.502	0.589
17		STANFORD	01_vs	33	0.480	0.800	0.343					
		AJ-SEVAL	02_second	34	0.477	0.618	0.389					
	13	YAMRAJ	02_lexical	35	0.470	0.677	0.360	24	0.363	0.511	0.508	0.514
18		WHUHJP	02_whuhjp	36	0.425	0.299	0.731					
		WHUHJP	01_whuhjp	37	0.387	0.275	0.651					
		BASELINE	**random**	**38**	**0.266**	**0.192**	**0.434**	**28**	**0.017**	**0.350**	**0.215**	**0.949**

Appendix C

The preliminary experiment materials are as follows:

- The video description form can be found on this link: https://docs.google.com/forms/d/19XgM3VvsLkpUGkKz4u_TbdQP8YnMa3vhyThwe5tlvzk/viewform
- The guidelines are for annotators, written in Turkish, are given below, followed by a summary-translation of the guidelines in English.

I. Türkçe Anlamsal Benzerlik Derlemi için Açıklama Rehberi

"PARAPHRASE" NEDIR?

"Paraphrase" kelimesi İngilizce bir kelime olup, Türkçe 'de tek bir kelimeye karşılık gelmemektedir. Bu kelime, Türkçe 'de "**aynı olayı farklı kelimelerle ifade etmek**" olarak açıklanır. Cümle olarak baktığımız zaman ise bir cümlenin anlamını bozmadan farklı kelimelerle ifade edebilmektir. Bir cümlenin başka bir cümleye dönüşmesi kimi zaman es anlamlı kelimeler kullanılarak ve/veya cümlenin ögelerinin yer değiştirmesi vb. gibi değişimlerle elde edilir. Örneğin:

Cümle 1: Edirne Valiliği şehir merkezindeki başıboş atlar için 5 bin lira maaşla çoban tuttu.

Cümle 2: Edirne'de valilik başıboş atlar için aylık 5 bin liraya çoban tuttu.

Yukarıdaki iki cümlede kelimelerin yerleri (yani cümledeki görevleri değişmiş) ve "maaş" kelimesi "aylık" anlamında kullanılmış. Ama cümleye bütün olarak baktığımızda anlamsal olarak tamamen aynıdır.

Cümleleri puanlamadan önce bu olayı uygulamalı olarak görmek için size gönderdiğim formun sonuçlarına bakalım. Aynı video için yapılan tek cümlelik açıklamalar, sizin açıklamanızla birlikte aşağıda verilmiştir:

VIDEO 1	
1	Bir kuş, mutfak lavabosu içinde musluktan akan suyun altına girmeye çalışıyor.
3	Kuş lavobada su ile oyun oynuyor.
4	Muhabbet kuşu lavabonun içinde hem temizleniyor, hem de oyun oynuyor.
5	Kuş lavaboda suyun tadını çıkartıyor.
6	Kuş lavaboda çeşmeden akan suyun altında yıkanmaya çalışıyor.
7	Kuş, suyun altında ıslanıyor.
8	Kuş , suyla dans ediyor.
VIDEO 2	
1	"South African" firmasına ait bir uçak gökyüzünden yere inmek için çeşitli hareketler yapıyor.
3	Uçak havada defalarca dönmüştür.
4	Yolcu gemisi gökyüzünde şov yapıyor.
5	Uçak bazı sorunlardan dolayı iniş yapmakta zorlanıyor.
6	Uçak iniş yapmak için pistin müsait olmasını bekliyor.
7	uçak piste iniş yapamıyor.
8	uçak havada ahenkle dans ediyor

“Paraphrase” tanımına geri dönecek olursak; bu çalışmada yer alacak 5 farklı kişi de aynı videoyu tek cümleyle açıkladı ama görmüş olduğunuz gibi her bir cümle birbirinden farklı. Ayni olayi anlatıyor olsa bile, cümleler ayni anlama gelmeyebilir: Uçak videosunun 2.cumlesinde olay oldukça detaylı anlatılmışken, 5.cumle olayı en sade halinde anlatmıştır. 2 ve 5 anlamsal olarak ayni değildir, ama anlamsal olarak yakındır. 4. Cümle ise “uçak” kelimesi yerine “yolcu gemisi” ifadesini kullanmıştır. 1. Ve 5. cümlelerde ise “South African firmasına ait bir uçak” ve “South African uçağı” özneleri birbirinin yerine kullanılmıştır. 3. Cümle ise sadece uçağın havada dönmesi kısmına odaklanmıştır.

Tıpkı bu video çalışmasında olduğu gibi; farklı haber siteleri/gazeteler günlük olan olayları, aynı olay olsa bile farklı şekilde ifade etmektedir.

VERI TIPI: Veri, cümle çiftlerinden oluşmaktadır. Bu cümleler Türkçe haber sitelerinden günlük haberlerden toplanmıştır ve cümleler, ayni konudan bahseden farklı haber sitelerinden alınan cümlelerin eşleştirilmesiyle derlenmiştir. Cümle çiftlerinin önemli özelliklerinden biri ortak kelimeler taşımasıdır.

GOREV: Ayni gün haber olmuş, ayni olay anlatan cümle çiftlerinin birbirine anlamsal olarak benzeyip benzemediğine karar verip, benzerliğin derecesini aşağıdaki sisteme göre puanlamak.

PUANLAMA: Puanlama dereceleri 0-5 arasında olup; her bir derecelendirme için, bir cifti örnek cümle verilip, aşağıdaki tabloda açıklanmıştır.

NOT: Cümlelerde yazım ya da noktalama yanlışları ve bunun gibi hatalar olabilir. Cümlelerin orijinalliğini bozmamak adına cümlelerde herhangi bir değişiklik yapılmamıştır. Eğer bu yanlışlar anlamı bozmuyorsa olduğu gibi değerlendirilmelidir.

5- AYNI ➔ Bu iki cümle tamamıyla ayni olayı anlatmakta, ayni bilgiyi içermektedir.
Cümle 1: ASELSAN yeni geliştirdiği Torpido Karşı Savunma Torpidosu (TORK) ile tekerlekli ve paletli kara platformlarının temelde zırhlı tanklara karşı savunmasını sağlamak üzere geliştirmiş olduğu yeni nesil tanksavar füze sistemini ziyaretçilerle ilk kez buluşturuyor. **Cümle 2:** Torpido 'ya karşı Tork Savunma teknolojileri alanında Türkiye'nin en büyük kuruluşu olan ASELSAN, geliştirilen harp sistemlerinin yanı sıra tekerlekli ve paletli kara platformlarının temelde zırhlı tanklara karşı savunmasını sağlamak üzere geliştirilen yeni nesil tanksavar füze sistemini ilk defa sergiliyor. **Açıklama:** Bu cümle çiftinde, ögeler yer değiştirerek "ASELSAN'ın yeni ürettiği TORK adli teknoloji" anlatılmış; bu teknolojinin 'ilk kez insanlar karşısına çıktığı' farklı kelimelerle ifade edilse bile anlamı değiştirmemektedir.
4-YAKIN ➔ Bu iki cümle anlamsal olarak çok yakındır; fakat bazı <u>önemsiz (anlamı değiştirmeyen)</u> detaylar farklıdır.
Cümle 1: Dünyanın en büyük 4'üncü asma köprüsü özelliği taşıyan köprünün kule montajı, 252 metrede tamamlandı. **Cümle 2:** Dünyanın en büyük orta açıklığa sahip 4'üncü asma köprüsü olan İzmit Körfez Geçişi Asma Köprüsü'nün kule montajı, 252 metrede tamamlandı. **Açıklama:** Bu iki cümle tamamen ayni olayı anlatır: "asma köprünün kurulması" ; fakat 1.Cumle'de köprünün ismi verilmemişken, 2.Cumle'de ismiyle birlikte "en büyük orta açıklığa sahip" bilgisi de verilmiştir. Anlamı değiştiren bir detay içermemektedir.
3- ALAKALI ➔ Bu iki cümle anlamsal olarak benzerlik taşır, fakat <u>önemli (anlamı değiştiren)</u> detaylar farklıdır.

Cümle 1: AK Parti'nin yüzde 42,5 oranında gözüktüğü anketin en çarpıcı sonucu ise HDP 'nin oy oranında gözlendi.

Cümle 2: AK Parti'nin oy oranın yüzde 43 olduğunu söyleyen Bayrakçı HDP 'nin ise baraj sınırında olduğunu kaydetti.

Açıklama: Bu cümle çiftinde, AKP ve HDP arasındaki oy oranı karşılaştırmış ve anlamsal olarak benzemektedir; fakat 1.Cumle de "HDP 'nin oy oranın çarpıcı olduğu" bilgisi ile "baraj altında olduğu" bilgisi anlamsal olarak uyuşmaz. Çünkü "oy oranın çarpıcı olması" yorumdur ama baraj altında olması kaynağa dayalı bilgidir.

2 – KISMEN ALAKALI ➔ Bu iki cümle anlamsal olarak benzememektedir fakat bazı ortak detaylar bulunmaktadır.

Cümle 1: 7 Haziran seçimleri öncesi HDP için düzenlenen şarkı için Feridun Düzağaç twitter hesabından açıklama yaptı.

Cümle 2: HDP 'nin 7 Haziran seçimleri öncesi kullandığı Feridun Düzağaç'ın 'F.D' isimli parçasını izinsiz aldığı ortaya çıktı.

Açıklama: Bu iki cümle anlamsal olarak benzemez çünkü 7 Haziran seçimlerindeki Feridun Düzağaç şarkisi ortak detay olmasına rağmen, 1. Cümle bu şarki için Feridun Düzağaç' in tweetinden, 2. Cümle ise bu şarkinin HDP tarafından izinsiz alındığı bilgisini vermektedir.

1-ICERIK ➔ Bu iki cümle kesinlikle farklı anlamdadır fakat ayni konudan bahsetmektedir.

Cümle 1: Muğla Valisi Amir Çiçek, vinçle Dalaman Hava Meydan Komutanlığına nizamiyedeki bariyerleri kırarak giren sürücünün ölmesine ilişkin, "Olayın terör bağlantısı olduğunu düşünmüyoruz" dedi.

Cümle 2: Muğla'da, vinçle Dalaman Hava Meydan Komutanlığına nizamiyedeki bariyerleri kırarak giren sürücü, askerlerin ateş açmasının ardından, aracın kanala devrildiği olayda öldü.

Açıklama: Bu cümleler kesinlikle ayni olayın farklı sonuçlarından bahsetmektedir. 1.Cumle'de olayın terör bağlantısından bahsedilmişken; 2.cumle kişinin kazada öldüğünü anlatmıştır.

0- ALAKASIZ ➔ Bu iki cümle <u>hem anlamsal hem de konu açısından</u> tamamen birbirinden farklıdır.

Cümle 1: AK Parti Balçova eski İlçe Başkanı Oktay Duru, partisinden istifa edeceğini açıkladı.

Cümle 2: Mersin'de AK Parti Toroslar İlçe Başkanı Bilal Babalıklı, yaklaşık 4 yıldır sürdürdüğü görevinden istifa etti.

Açıklama: Bu cümleler iki farklı anlamsal olarak tamamen farklıdır, iki farklı insanın istifalarından bahsetmiş. "İstifa" sadece kelime benzerliği gösterir, cümlenin bütününde anlamsal bir benzerlik yoktur.

Sonuçlar bilimsel bir çalışmada kullanılacağı için, sizden emin olarak puanlamanızı özellikle rica ediyorum. Emin olmadığınız cümleler için bir kez daha düşünün. Teşekkürler...

II. Turkish Paraphrase Corpus Annotation Guidelines

WHAT IS PARAPHRASING?

The word "paraphrase" is an English word and it cannot be translated to Turkish as one word. This word corresponds to the phrase "saying in other words" in Turkish.

Video Task Explanation

DATA TYPE: The data consist of sentence pairs. These sentences are collected from daily news websites and they are paired according to their similarity. One of the attributes taken into account during the collection process is that they have share common words.

TASK: It is to decide whether the two sentences -are taken from the same day news talking about the daily events- are semantically equivalent or the degree of their semantic equivalence.

SCORING: The scoring schema is chosen between 0-5; there are example sentence pairs are shown in below for the each degree along with their explanations. The sentences are in their original form so it is possible there will be spelling and punctuation mistakes. Please do not take into account any error unless they change the meaning of sentences. Before reading sample pairs, in order to provide a better understanding, we demonstrated the sentences you filled in the video form. This is to show that how everyone's view of point differs from each other for the same video snap.

	VIDEO 1
1	Bir kuş, mutfak lavabosu içinde musluktan akan suyun altına girmeye çalışıyor.
3	Kuş lavobada su ile oyun oynuyor.
4	Muhabbet kuşu lavabonun içinde hem temizleniyor, hem de oyun oynuyor.
5	Kuş lavaboda suyun tadını çıkartıyor.
6	Kuş lavaboda çeşmeden akan suyun altında yıkanmaya çalışıyor.
7	Kuş, suyun altında ıslanıyor.
8	Kuş , suyla dans ediyor.
	VIDEO 2
1	"South African" firmasına ait bir uçak gökyüzünden yere inmek için çeşitli hareketler yapıyor.
3	Uçak havada defalarca dönmüştür.
4	Yolcu gemisi gökyüzünde şov yapıyor.
5	Uçak bazı sorunlardan dolayı iniş yapmakta zorlanıyor.
6	Uçak iniş yapmak için pistin müsait olmasını bekliyor.

7	uçak piste iniş yapamıyor.
8	uçak havada ahenkle dans ediyor

5- IDENTICAL ➔ The two sentences are completely equivalent, as they mean the same thing.
Sentence 1: Develop new Defense Against Torpedo Torpedo is ASELSAN (torque) which was developed with a new generation of wheeled and tracked on the basis of land platforms to provide defense against armored tanks with anti-tank missile system brings together for the first time visitor. **Sentence 2:** Torpedo 'versus torque ASELSAN , Turkey's biggest company in the field of defense technologies, as well as the development of warfare as wheeled and tracked a new generation developed to ensure the defense basically against armored tanks on land platforms anti-tank exhibits for the first time the missile system.
4-CLOSE ➔ The two sentences are mostly equivalent, but some unimportant details differ.
Sentence 1: The world's largest 4th tower suspension bridge feature with the installation of the bridge was completed in 252 meters. **Sentence:** The world's largest mid-span suspension bridge with the 4th Izmit Bay Crossing Suspension Bridge tower installation was completed at 252 meters.
3- RELATED ➔ The two sentences are roughly equivalent, but some important information differs/missing.
Sentence 1: The most striking result of the survey appear AK 42.5 percent HDPE Party 's vote rate was observed . **Sentence 2:** AK Party's vote share of 43 percent that said Bayrakçı HDPE 's is noted that the threshold limits.
2 – CONTEXT ➔ The two sentences are not equivalent, but share some details.
Sentence 1: Feridun Duzagac made comments from his twitter account for the re-edited song that HDP used before the June 7 elections. **Sentence 2:** It appered that the Feridun Duzagac’ s song called “F.D.” used without permission by HDP before the June 7 elections.

1-SLIGHTLY RELATED ➔ The two sentences are not equivalent, but are on the same topic.
Sentence 1: Governor of Mugla , Dalaman Air Field Command related to the crane driver who died in the guardhouse breaking barriers , " We do not think the incident was a terrorist connection ," he said **.** **Sentence 2:** In Mugla, Dalaman Airfield crane driver Command breaking barriers in entering the main gate , after the soldiers opened fire , killing the vehicle overturned in the event that channel.
0- UNRELATED ➔ The two sentences are on different topics.
Sentence 1: AK Party Balçova former County Chairman Oktay Duru, has announced he will resign from the party. **Sentence 2:** Taurus Mersin AK Party District Chairman Bilal Babalik was resigned that continued for nearly 4 years .

III. Turkish Stop Words

acaba altmýþ altý ama bana bazý belki ben benden beni benim beþ bin bir biri birkaç birkez birþey birþeyi biz bizden bizi	bizim bu buna bunda bundan bunu bunun da daha dahi de defa diye doksan dokuz dört elli en gibi hem hep hepsi her hiç	iki ile mi ise için katrilyon kez ki kim kimden kime kimi kýrk milyar milyon mu mü mý nasýl ne neden nerde nerede	nereye niye niçin on ona ondan onlar onlardan onlari onlarýn onu otuz sanki sekiz seksen sen senden seni senin siz sizden sizi sizin	trilyon tüm ve veya ya yani yedi yetmiþ yirmi yüz çok çünkü üç þey þeyden þeyi þeyler þu þuna þunda þundan þunu